CO353: Computational Discrete Optimization

Winter 2011

Instructor: Chaitanya Swamy

MC 4033 ext. 33600
cswamy AT math DOT uwaterloo DOT ca

Time: MWF 11:30-12:20pm
Location: MC 4058

Office Hours: Wednesdays, 2-3pm, and **Apr 4, Apr 8, 2-3pm.**

TA: Cristiane Sato

MC 6219 ext. 35320
~~**Office Hours (updated):** Tuesdays, 4-5pm.~~
Cristiane is out of town and her office hours in the Apr 4-11 week are cancelled.

Announcements (in reverse chronological order)

- Apr 4: The **final exam** is scheduled for **Monday, April 11, 9-11:30am** in **RCH 205**. The final exam is **not comprehensive** and its **syllabus** includes everything covered from the lecture on Feb 11 to the end of the course. This includes the following topics:
  1. Computational complexity: P, NP, and NP-completeness.

2. Steiner trees and the primal-dual approximation algorithm for other network connectivity problems modeled by (proper or downwards-monotone) cut-requirement functions.
3. Location problems: set cover, and UFL with arbitrary and metric connection costs.

You will not be tested on the topics that were part of the midterm syllabus. However, you should know how to compute shortest paths and MSTs, as these form basic components of other algorithms (e.g., taking the metric completion of a graph and finding an MST in it).

- Apr 4: Here are the Solutions to Assignment 6.
- Apr 3: Here are the Solutions to Assignment 5.
- Mar 31: Some clarifications/corrections about Assignment 6.
  - Q1(b): assume that every element e is covered by some set in the collection.
  - Q2: in the description of the clustering step between parts 2(b) and 2(c), it should be "and remove k from L."
  - Q2(c): recall that c_{jk} is the shortest-path distance between clients j and k (given the $\{c_{ij}\}$ edge costs).

  The assignment has been upated to incorporate these clarifications/corrections.
- Mar 28: Assignment 6 (updated) is available. **Due on Monday, Apr. 4, 2011.**
- Mar 25: Some additional reference texts have been added to the "Books and other Supplementary Material" section.
- Mar 21: Here are the Solutions to Assignment 4.

- Mar 20: Assignment 5 is available. **Due on Monday, Mar. 28, 2011.**
- Mar 16: Here are the Solutions to Assignment 3.
- Mar 12: A crucial change has been made to Q1b) on Assignment 4. The correspondence between T-joins and feasible solutions to the network-connectivity problem tha tyou are asked to show is that every *minimal* feasible solution to the network-connectivity problem is a T-join and that every T-join is a feasible solution to the network-connectivity problem. The version posted above is the updated version.
- Mar 11: Assignment 4 (updated) is available. **Due on Friday, Mar. 18, 2011.**
- Mar 9: The slides used for the lectures on Mar 4 and Mar 7 have been posted in the "Topics Covered" section below (see the listing for Mar 7).
- Mar 2: Here are the Solutions to the midterm.
- Mar 1: Assignment 3 is available. **Due on Wednesday, Mar. 9, 2011.**
- Feb 16: Here are the Solutions to Assignment 2.
- Feb 14: The **syllabus for the midterm** is everything covered up to the lecture on Feb 8 and assignment 2.
  This includes shortest paths, MSTs, mininmum-cost arborescences, and matroids.
- Feb 2: Here are the Solutions to Assignment 1.
- Feb 1: The **midterm** is scheduled for **Thursday, February 17** from **5-7pm** in room **MC 4063**.
- Feb 1: Assignment 2 is available. **Due on Friday, Feb. 11, 2011.**
- Jan 28: **Clarification** about **Q3b) on**

Assignment 1. The proposed algorithm chooses a cut such that the subgraphs (A,E[A]) and (B,E[B]) are connected. (So the algorithm always returns a tree.)

- Jan 26: Since assignment 1 is due on Monday, just this week, Cristiane Sato will hold office hours on **Friday, Jan 28** from **2:30-3:30pm** in place her regular office hours on Monday, Jan 31 from 4-5pm. (Her regular office hours on Mondays from 4-5pm will resume from Feb 7.)
- Jan 22: Assignment 1 is available. **Due on Monday, Jan. 31, 2011.**
- Jan 22: Here are the Solutions to the Prerequisite-Quiz.
- Jan 22: There will be no class on Jan 24 as I will be out of town.
- Jan 15: I realized that the O(.)-notation (big-O notation) for specifying the asymptotic growth rate of functions is not something that is covered in MATH239. You can read up about this in Section 2.2 of [KT]. I may also post some notes online.
- Jan 10: The **Prerequisite-Quiz** will be held **in class on Monday, Jan 17**.
  The **syllabus** for the quiz is:
  1. **Basic graph notation, terminology, concepts**. Paths, cycles, graph connectivity, trees, spanning trees, bipartite graphs - Chapter 4, Sections 5.1-5.4 of the MATH 239 course notes (as of Fall 2008).
  2. **Simplex method**. Solving an LP using the simplex method (using a specified rule for breaking ties in the choice of entering and leaving variables) - Chapter 6 of the CO 350 course notes (as of Fall 2006).

3. **Duality**. Writing the dual of an LP, the complementary slackness (CS) conditions, using the CS conditions to determine if a given solution is optimal or not - Chapter 4 (except Section 4.7) of the CO 350 course notes (as of Fall 2006).

There will be 3 questions on the quiz.

- Jan 5: Welcome to CO 353.

Topics Covered

- Jan 5: Course Introduction.
- Jan 7: Shortest paths: Dijkstra's algorithm - Section 4.4 of [KT].
- Jan 10: Finished shortest paths. Defined the O(.)-notation for specifying the asymptotic growth rate of functions and looked at some examples.
- Jan 12: Minimum spanning trees (MSTs): Prim's algorithm - Section 4.5 of [KT].
- Jan 14: Finished Prim's algorithm. Kruskal's algorithm - Sections 4.5, 4.6 of [KT].
- Jan 17: Prerequisite Quiz.
- Jan 19: Finished Kruskal's algorithm. A clustering application - Section 4.7 of [KT].
- Jan 21: Finished clustering application. Shortest paths modeling: trucking problem.
- Jan 24: No class.
- Jan 26: Started minimum cost arborescences - Section 4.9 of [KT].
- Jan 28: Min-cost arborescences continued: Edmonds' algorithm.
- Jan 31: Finished min-cost arborescences.
- Feb 2: Started with matroids: definition and examples.
- Feb 4: Matroids continued: maximum-weight independent set problem, the greedy

algorithm, and MSTs.

- Feb 8: Finished matroids: proved the correctness of the greedy algorithm. Stated the matroid intersection problem and gave some examples.
- Feb 11: Introduction to computational complexity - Chapter 8 of [KT]. Decision problems and polynomial-time reductions - Section 8.1 of [KT].
- Feb 14: The classes P, NP and NP-completeness - Section 8.3 of [KT]. Showed that Steiner tree and 3-SAT are in NP.
- Feb 16: Proved that Set Cover is NP-complete via a reduction from 3-SAT - Sections 8.1, 8.2 of [KT].
- Feb 18: Proved that Steiner tree is NP-complete by reducing Set Cover to it. Sections 8.4-8.7 of [KT] contain many more NP-completness proofs via reductions. Gave a 2-approximation algorithm for minimum Steiner tree based on shortcutting an MST in the metric completion of original instance - Section 3.1 of [V].
- Feb 21-Feb 25: No class due to Reading Week.
- Feb 28: Introduced the general paradigm for the design and analysis of approximation algorithms. See Section 1.1 of [V]. Talked about how linear programming (LP) relaxations of integer programs can be used to obtain lower bounds. Described the LP relaxation of the Steiner tree problem and obtained the dual LP. Proved weak duality, and stated strong duality, complementary slackness (CS) conditions.
- Mar 2: Discussed the two common ways in which LPs are used in approximation algorithms design and analysis: (i) primal-dual

algorithms, (ii) LP-rounding algorithms. See Chapter 12 of [V] for a discussion of these and related topics like integrality gap.

- Mar 4: Gave combinatorial and economic intuition for the dual lower bound for the Steiner tree LP. Stated the various design rules used to obtain the primal-dual algorithm for Steiner tree.
- Mar 7: Decribed the primal-dual algorithm for Steiner tree. Here are the slides used for some of this lecture and the previous one.
  Proved a basic fact about minimal violated sets. Stated a certain key lemma, which can be viewed as a suitable relaxation of the CS conditions, and started with the proof of 2-approximation of the primal-dual algorithm assuming this lemma. See Chapter 22 of [V]. This chapter considers a more general problem, the Steiner forest problem, which we will cover in a later lecture.
- Mar 9: Finished proof of 2-approximation. Proved the key lemma stated in last lecture. Started with modeling network connectivity problems with 0-1 proper functions.
- Mar 11: Finished proof that the primal-dual algorithm for network connectivity with 0-1 proper functions is a 2-approximation algorithm.
  The primal-dual schema for network connectivity problems is due to Goemans and Williamson. Here is a survey by them (Chapter 4 in the book "Approximation Algorithms for NP-Hard Problems" listed below). This contains much more than what we covered in class and is at a little higher level. Sections 4.1, 4.3-4.6 are particularly relevant. The survey also contains some excercises.

Here is their original paper: "**A General Approximation Technique for Constrained Forest Problems**", *SIAM Journal on Computing*, 24:296-317, 1995.
Started the set cover problem. Described the greedy algorithm for set cover and wrote down the LP-relaxation for set cover - Section 2.1 of [V]. This section also analyzes this greedy algorithm without any reference to a (primal or dual) LP.

- Mar 14: Gave a primal-dual interpretation of the greedy algorithm and started with the analysis that this greedy algorithm is an H_Δ-approximation algorithm analyzed using dual fitting, where Δ is the maximum size of a set and H_k is the k-th harmonic number - Section 13.1 of [V].
- Mar 16: Finished the analysis of the greedy algorithm. Gave a primal-dual B-approximation algorithm, where B is the maximum number of sets an element appears in, based on finding a maximal dual feasible solution and then picking the sets that are fully paid for. Showed that a maximal dual feasible solution can be constructed by a greedy rule - Section 15.2 of [V].
  Started with LP-rounding algorithms for set cover - Chapter 14 of [V]. Described a basic threshold LP-rounding algorithm.
- Mar 18: Gave a threshold LP-rounding B-approximation algorithm - Section 14.1 of [V]. Started with randomized rounding for set cover.
- Mar 21: Gave an $O(\ln n)$-approximation algorithm for set cover using randomized rounding - Section 14.2 of [V].
- Mar 23: Showed how one can slightly modify the randomized-rounding algorithm to obtain

an algorithm that always returns a set cover and has the same *O(ln n)* performance guarantee. Started with the uncapacitated facililty location (UFL) problem with arbitary assignment costs. Gave an LP relaxation, and economic motivation for the dual LP - Section 24.1 of [V].

- Mar 25: Described an LP-rounding algorithm for UFL with arbitrary assignment costs that uses duality and complementary slackness to reduce UFL to set cover.
- Mar 28: Completed the analysis of the LP-rounding algorithm for UFL with arbitrary assignment costs. Showed that an LP-based α-approximation for set cover gives an $(\alpha+1)$-approximation for UFL. Started with UFL with metric assignment costs (metric UFL). Stated an LP-rounding algorithm based on clustering facilities around certain "hub"-clients.
- Mar 30: Proved that the LP-rounding algorithm is a 4-approximation algorithm via complementary slackness. Started with the Jain-Vazirani (JV) primal-dual algorithm for UFL - Chapter 24 of [V].
- Apr 1: Completed the description of the JV algorithm and started with the analysis.
- Apr 4: Completed the analysis of the JV algorithm and proved that it attains an approximation ratio of 3.

Course Overview

An introduction to combinatorial and discrete optimization. Discrete optimization problems have the common property that the underlying decisions are (or can often be modeled as) {0,1}-decisions. This course will consider a variety of discrete

optimization problems focusing on (1) modeling aspects, showing how combinatorial optimization problems can be used to abstract problems arising in a wide variety of contexts and applications; (2) algorithmic issues, where we will study various techniques for the design and analysis of algorithms for solving (exactly or approximately) combinatorial optimization problems. Some broad classes of problems that we will consider are:

(a) Network connectivity problems, which typically entail constructing a minimum-cost subgraph of an underlying
network that possesses certain desired connectivity properties. Examples include shortest paths, minimum spanning
trees, minimum Steiner trees, generalized Steiner forests;

(b) Location problems, that deal with the optimal placement of "facilities" at the nodes of a network. A prototypical
example is the uncapacitated facility location problem;

(c) (General) Integer programming problems, which are a rather broad class of problems that encompass the two classes above.

The algorithmic techniques that we will explore via the lens of discrete optimization problems include greedy algorithms, dynamic programming, approximation algorithms, and cutting-plane and branch-and-bound methods for general integer programming.

Prerequisites

The official prerequisites are: MATH 239/249 and

CO 250/CO 350/CO 352/CM 340/CO 355.
If you do not satisfy these prerequisites but are interested in taking this course for credit, come and talk to me.

I will assume that you have some basic knowledge about graphs like basic terminology, notation, concepts and algorithms (some pointers to refresh your memory about graphs are given below). I will devote a lecture or two to linear programming and duality (in the context of network design problems), which should be accessible even if you have never seen linear programs before. A background in algorithms might be useful but is not essential.

Assignments and Exams

There will be 5 or 6 assignments that will be handed out roughly once every two weeks, and will be due in one or two weeks time. Late assignments will not be accepted. You are allowed to discuss the assignments with each other, but **you must write up the final solution individually**. You must also write down the names of all your collaborators on the assignment (this does not affect your mark).
There will (as usual) be a midterm and final for the course. Also, there will be a prerequisite-quiz held in the second week of classes. The weightate for the assignments and the exams are tentatively as follows:

Prerequisite-quiz:	4%
Assignments:	24%
Midterm:	32%
Final:	40%

Books and other Supplementary Material (updated)

There are no course notes or prescribed textbook for this course.
Some of the topics covered in the course can be found in the following texts. (This collection will be updated as necessary.)

- *Algorithm Design*, J. Kleinberg and É. Tardos. Addison-Wesley, 2005.
  See Chapters 2 and 3 for basics on algorithm analysis and graphs. Referred to as [KT] in Topics Covered.

- *Approximation Algorithms*, Vijay Vazirani. Springer-Verlag 2001. Referred to as [V] in Topics Covered.

Portions from these texts relevant to the material covered in lecture, and possibly links to some online sources will be posted in the Topics Covered section from time to time.

Some other books that might be useful are:

- *Introduction to Algorithms*, T. Cormen, C. Leiserson, R. Rivest, and C. Stein. McGraw-Hill 1990.
  A standard reference book covering a wide variety of topics.

- *Approximation Algorithms for NP-Hard Problems*, D. Hochbaum, PWS Publishing Company, 1997.
  A more advanced-level reference book on approximation algorithms.

Cheating Policy

Students suspected of cheating will be reported to the associate dean for their actions. They will receive no credit for the assignment in question, and in addition any penalty that the associate dean's office prescribes. Cheating on assignments includes copying others' work, including past course solutions and external resources found online or in books, and submitting it as your own. As mentioned above, students are allowed to discuss the assignments with each other, but **they must write up the final solution individually**. They must also write down the names of all your collaborators on the assignment (this does not affect their mark).