

# Winter 2008 - COSC 105: Algorithms and Data Structures

## Staff

<b>Instructor:</b>	<b>Teaching Assistant:</b>
Lisa Fleischer	Umang Bhaskar
lkf@cs.dartmouth.edu	umang@cs.dartmouth.edu
Sudikoff 206	Sudikoff 205

## Time and Place

TTh 2-3:50, Sudikoff 214 or 115 (!)

X-hour: W 4:15 - 5:20. We may occasionally use X-hour for extra classes. If so, you will be given notice the previous week.

## Office Hours

Lisa: Wednesday 2 - 3, or by appointment (email).

## Sources

### Textbooks (recommended, not required)

J. Kleinberg and E. Tardos. *Algorithm Design*. Addison-Wesley, 2005.

D. C. Kozen. *The Design and Analysis of Algorithms*. Springer-Verlag, 1991.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, 1979.

### Other References

T. Cormen, C. Leiserson, R. Rivest, C. Stein. *Introduction to Algorithms*. McGraw Hill,

V. Vazirani. *Approximation Algorithms*. Springer-Verlag.

R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge Press.

C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall.

Most of these titles will be on reserve in Baker-Berry Library.

## Handouts

Supplementary material including copies of some lectures from Kozen's book will be made available on an as needed basis either in class or via Blackboard.

## Homeworks and Exams

There will be mostly weekly homeworks consisting of about 2 problems. You may collaborate on the homework in groups of 2 or 3. If you collaborate, submit one copy of your solutions with the names of all collaborators. Acknowledge all sources, including others in the class from whom you obtained ideas. Please type or write legibly and staple. Solutions are due by hard copy on day and time stated on homework. There will be a point deduction of 10% per day for unexcused late homework. Assignments and solutions will be posted on Blackboard. No late homeworks will be accepted after solutions are posted.

There will be two take home exams. The first will be scheduled sometime between Feb 7 and Feb 14. The second one is during exam period. No collaboration is allowed on the exams. More details to come.

Approximate weights: Homework 50%, Exams 25% each.

More on the homework:

The TA for the class is also taking the class, so each person in class must submit to the instructor by email a **codename**. Then on your homework or exams, put your codename, not your real name. The TA will not know the real names, only the code names.

We encourage you to work in small groups. It is more fun for you because you have others to bounce ideas off and you learn more, and it is more fun for us because there is less to grade. Only one homework is submitted per group, and all code names must be listed at top.

The preferred interpretation of *collaboration on the homework* is not "Alice does problem 1 and Bob does problem 2," but rather "Alice and Bob do problem 1 and Alice and Bob do problem 2". This way, you learn more.

Whenever you give an algorithm, always give a correctness proof.

Give an analysis of the complexity of your algorithm if that is the point of the question.

Super formal proofs are not necessary, a good convincing argument will do. Your proofs should be crystal clear. We should not have to figure out how or why leaps of logic are true, even if they are. If you do not understand a step, it is better to say so. Use figures if they help exposition. Avoid pseudo-code, a high-level explanation of an algorithm is better. If you must use pseudo-code, comment liberally. Please keep your solutions short and to the point.

Staple your solutions.

## Prerequisites

It is assumed you have taken a basic algorithms course that required some proofs. This should have included:

Discrete math structures including graphs, trees, dags;

Basic data structures such as lists, trees, heaps, sorting, and searching;

Asymptotic complexity,  $O()$  notation and  $o()$  notation;

NP-completeness and polynomial-time reducibility;

Basic algorithms: DFS, BFS, shortest paths, divide-and-conquer,

In general, I will be assuming familiarity with the first three chapters of KT plus some of the early sections of other chapters (4, 5, 6, 7, 8) in KT.

## Approximate Syllabus

We will cover some basic algorithmic paradigms and techniques, as time permits. This list is subject to change, and we will may not have time to cover everything. The emphasis will be on proving correctness and asymptotic run time.

**Greedy algorithms:** spanning trees, Steiner trees, matroids, arborescences, multicast cost sharing. (Kozen Lec 1-3, KT Ch 4)

**Advanced data structures:** advanced heaps and self-organizing data-structures. Amortized analysis. (Kozen 8-13)

**Network Flows:** maximum flows and minimum cuts, the preflow-push algorithm, minimum cost perfect matching, multicommodity flows, applications to image segmentation. (KT Ch 7, Kozen Ch 14, 16-20)

**Dynamic Programming:** basics, dynamic programming on trees, tree decompositions, and algorithms for graphs with bounded tree-width (KT Ch 5)

**NP-completeness:** (Kozen Ch 21-24, KT Ch 8, Garey & Johnson)

**Approximation Algorithms** (KT Ch 11)

**Local Search** (KT Ch 12)

**Randomized Algorithms** (KT Ch 13)