Computer Science 30b
Algorithm Design and Analysis
Course Outline – Spring Term 1991

Course instructor: Harry Mairson (`mairson@chaos`), 132 Ford (736-2724). *Office hours:* 2–4pm Tuesday and by appointment.

Teaching assistants: Adam Weitzman (`weitzman@chaos`) and Aline Yurik (`aline@chaos`), 7 Ford-Sydeman (736-2739).

Time and place: M W F 4–5pm, Ford 229.

Recommended Text

Thomas Cormen, Charles Leiserson, and Ronald Rivest, **Introduction to Algorithms.** MIT Press/McGraw Hill, 1990.

Also recommended (in library, or borrow my copy):

Alfred Aho, John Hopcroft, and Jeffrey Ullman, **The Design and Analysis of Computer Algorithms.** Addison Wesley, 1974.

Robert Sedgwick, **Algorithms**. Addison Wesley, 1987.

David Harel, **Algorithmics: The Spirit of Computing**. Addison Wesley, 1987.

Michael Garey and David Johnson, **Computers and Intractibility: A Guide to the Theory of NP-Completeness**. Freeman, 1979.

Course Description

This course will cover basic techniques for designing and analyzing algorithms, as well as examining the "intrinsic difficulty" of computational problems for which algorithms are proposed. We'll follow many of the topics in Cormen, Lieserson, and Rivest (CLR), covering the classical "core" topics on algorithms. Depending on the background of the class (how much did you learn in CS21b, anyways?), we may branch off into more advanced material.

How Hard Will This Course Be?

This course will not be easy. On the other hand, it won't be an utter killer either. Our emphasis will be on a mathematical understanding of algorithms rather than practical questions relating implementation. I expect you to have taken CS21b and not be afraid of discrete math and a little algebra–if nothing else, maybe this course will convince you that discrete is not totally boring and irrelevant to computer science (one out of the two, anyways). As always, that good old "mathematical maturity" goes a long way. I will not follow the text religiously, but it will provide a good guide.

What Work Will Be Required?

There will be problem sets assigned about every two weeks, usually between four and six problems. There will also be a midterm and a final. The grades will be computed on the basis of 50 percent on homework and 25 percent on each of the exams, with a suitable fudge factor (a variable constant) to make sure that Justice is done.

Tentative Course Outline

Introduction

- Models of computation, design of efficient algorithms.
- Examples: Resources (gossips) and correctness (stable marriage).
- Asymptotics (O, o, Ω, ω). *(CLR 2.1–2.2)*
- Divide and conquer. *(CLR 4.1–4.3)*
- Dynamic programming. *(CLR 16.1–16.3)*

Sorting and searching, upper and lower bounds

- Comparison tree model *(CLR 9.1)*
- Worst-case and average-case lower bounds on sorting *(CLR 9.1)*
- Lower bounds on finding maximum, second largest, maximum and minimum, median
- Analysis of heapsort and quicksort *(CLR 7, 9)*

Set manipulation

- Union-find bounds. Applications. *(CLR 22)*
- Vuillemin's binomial queue *(CLR 20)*
- Red-black trees *(CLR 14)*

Graph algorithms

- Minimum spanning tree *(CLR 24)*
- Depth-first search *(CLR 23.3)*
- Strongly connected components *(CLR 23.5)*
- Transitive closure and matrix multiplication *(CLR 26.4)*

Amortized analysis *(tentative!)*

- Fundamentals *(CLR 18)*
- Application: Self-adjusting binary trees
- Application: Fibonacci heaps *(CLR 21)*

The Fast Fourier Transform and Polynomial Multiplication *(CLR 32)*

NP-Completeness and computational complexity *(CLR 36, 37)*

- Cook's Theorem
- Proof techniques
- approximation algorithms

Recommended reading, additional references

Here's an abridged bibliography of relevant books you may want to go to if you're stumped on something, or doing a library search for the solution to some homework exercise, or you have nothing better to do, or you're looking for some impressive fat books with smooth creamy pages for your coffee table, to impress friend and foe alike.

The following are recommended for mathematical "Knuthian" analysis of algorithms. You can probably get them at any good bookstore (Quantum Books in Kendall Square, Harvard or MIT Coop). Or you can borrow them from me!

Donald Knuth, **The Art of Computer Programming, Volume I: Fundamental Algorithms**. Addison Wesley, 1973.

Daniel Greene and Donald Knuth, **Mathematics for the Analysis of Algorithms.** Birkhauser, 1982.

Ronald Graham, Donald Knuth, and Oren Patashnik, **Concrete Mathematics**. Addison Wesley, 1988.

Paul Purdom and Cynthia Brown, **The Analysis of Algorithms**. Holt Rinehart and Winston, 1985.

Two books on graph and combinatorial algorithms that might be worth a look:

Christos Papadimitriou and Kenneth Steiglitz, **Combinatorial Optimization: Algorithms and Complexity**. Prentice-Hall, 1982.

Edward Reingold, Jurg Nievergelt, and Narsingh Deo, **Combinatorial Algorithms: Theory and Practice**. Prentice-Hall, 1977.

And while we're mentioning Great Books of computer science, some excellent surveys of algorithms, design, analysis, correctness, etc.:

Robert Sedgwick, **Algorithms**. Addison Wesley, 1987.

David Harel, **Algorithmics: The Spirit of Computing**. Addison Wesley, 1987.

Edsgar Dijkstra and W. H. J. Feijen, **A Method of Programming.** Addison Wesley, 1988.