Introduction to Analysis of Algorithms	**Approximate Syllabus**
CS 482 Spring 2003	**January 20, 2003**

This list of topics is meant to convey the course organization and the range of subjects covered in the course. With each topic we give a list of problems, applications, and sub-topics that we will discuss. The details are subject to change.

- Introduction
  - An opening problem: stable marriages
  - Range of problems we will consider
- General Algorithmic Techniques
  - Greedy algorithms
    - Finding optimal solutions with greedy methods
      (scheduling time intervals)
    - The minimum spanning tree problem
  - The Divide and Conquer method
    (some basic primitives in computational geometry)
  - Dynamic Programming with many applications
    - weighted interval scheduling
    - knapsack problems
    - shortest paths
    - sequence alignment
      (including efficient implementation via divide and conquer)
  - Flows and Cuts in Networks
    - The basic flow and cut problems
    - Basic methods: augmenting paths
    - Application to matching
    - Polynomial time methods
    - Extensions to more general models
    - Applications to resource allocation, sequencing, and segmentation.
- Computational Intractability
  - NP-completeness
    hardness of problems in optimization and constraint satisfaction.
    - How to show NP-completeness: reducibility

    * Examples including the traveling salesman problem, 3-dimensional matching, covering, packing, partitioning problems, and subset sum.
  - PSPACE completeness
    hardness of problems in artificial intelligence and game-playing.

- Algorithms for Hard Problems
  - Improved exponential methods
  - Approximation algorithms
    * greedy algorithms
      (load balancing, facility location)
    * The application of linear programming
  - Local search techniques
    * The Metropolis Algorithm
    * Simulated annealing
    * Applications to graph partitioning and neural networks

- Randomized algorithms
  (including contention-resolution protocols and satisfiability heuristics)

- Algorithms that run forever