CSE 417: Algorithms and Computational Complexity

Autumn 2002
Instructor: Paul Beame

Catalog Description:
- Design and analysis of algorithms and data structures.
- Efficient algorithms for manipulating graphs and strings.
- Fast Fourier Transform.
- Models of computation, including Turing machines.
- Time and space complexity.
- NP-complete problems and undecidable problems.

What the course is about

Design of Algorithms
- design methods
  - dynamic programming
  - divide and conquer
- common or important types of problems
- how to analyze algorithms
  - resource usage

Computability
- Turing machines and ideal computers
- What kinds of problems can computers solve?
  - Are there any well-defined problems that computers can’t solve?

Complexity and NP-completeness
- simply being able to solve problems in principle is not enough
  - algorithms must be efficient, too
- NP
  - wide class of useful problems whose solutions can be easily checked (but not necessarily found) efficiently
  - NP-completeness
    - useful for understanding when problems are hard to solve

Cryptography (e.g. RSA, SSL in browsers)
- Secret: $p, q$ prime, say 512 bits each
- Public: $n$ which equals $p \times q$, 1024 bits

In principle
- there is an algorithm that given $n$ will find $p$ and $q$ by trying all $2^{512}$ possible $p$’s.

In practice
- security of RSA depends on the fact that no efficient algorithm is known for this
Algorithms versus Machines

- We all know about Moore’s Law and the exponential improvements in hardware but that’s not the whole story…
- e.g. solving sparse linear equations over past few decades
- 10 orders of magnitude improvement in speed
  - 4 orders of magnitude improvement in hardware
  - 6 orders of magnitude improvement in algorithms

Course Staff

- Instructor: Paul Beame beame@cs
  - Office: Sieg 416  Hours: TBA
  - Phone: 543-5114
- TAs:
  - Deepak Verma deepak@cs
  - Michael Nelson nelsonmj@cs
  - Office: Sieg 226  Hours: TBA

Course Web Page

- Go to CSE home page
  - http://www.cs.washington.edu/
- Follow links at left to
  - Education → Course Home Pages → CSE 417
- or go directly to
  - http://www.cs.washington.edu/education/courses/cse417\textunderscore 02au

Class Mailing List

- cse417@cs.washington.edu.
  - Follow the link in the left column on the course web page to sign up
  - Everyone is expected to be reading cse417 e-mail to keep up-to-date on the course.

Textbook

- The ALGORITHM design manual by Steven Skiena,
  - published by Springer-Verlag.
  - In addition, I will borrow a small amount of material from Introduction to the Theory of Computation by Michael Sipser
- Copies should be on reserve in the Engineering Library.

What you’ll have to do

- Written homework assignments (about 4)
  - English exposition and pseudo-code
  - Analysis and argument as well as design
- 1-2 programming assignments
  - Documentation and analysis as well as working code
- Midterm  Friday, November 8 in class
- Final Exam  December 18, 8:30-10:20
Rough Division of Time

- Turing Machines & Computability (1 week)
- Algorithms (7 weeks)
  - Analysis of Algorithms
  - Basic Algorithmic Design Techniques
  - Graph Algorithms
  - (Fast Fourier Transform)
  - Pattern Matching & Finite Automata
- Complexity & NP-completeness (2 weeks)

Reading Assignment

- Imagine you lived in the early 1900’s in the days before any machine we would call a computer existed...
- …when ‘computers’ were people who did actuarial calculations for insurance and trajectory calculations for naval gunnery and then…
- read the handouts with the excerpts of the papers of Turing and Post on what an ideal computer would be.