IE426: Algorithms in Systems Engineering: Lecture 1

Jeff Linderoth

Department of Industrial and Systems Engineering
Lehigh University

January 15, 2007

Today's Outline

- About this class.
- About me
- About you
  - Say Cheese!
- Quiz Number 0
- Background in Algorithms

Class Overview

- Meeting Times: Monday, Wednesday, Friday 11:10AM-12PM, 375 Packard Lab.
- Office Hours: (*Please* try to use them).
  - Monday 10:00-11:00
  - Wednesday 10:00-11:00
  - By Appointment (610-758-4879)
- Course HomePage:
  - `http://www.lehigh.edu/~jtl3/teaching/ie170`
  - I will post outlines of lecture notes there before class.
  - But not necessarily *much* before.
- Syllabus dates are somewhat tentative

Course Details

- Learning is better if you participate.
  - I will call on you during class. (Gasp!)
  - A portion of your grade depends on your participation.
- Labs and Problems Sets
  - The best way to learn is by doing. $\Rightarrow$ I give lots of homework.
  - Don't be late! 10% Grade penalty for every late day. No exception!
- Quizzes: Scheduled on an "as needed" basis. If I sense you are working independently and making good progress on the labs and problems sets, we will have fewer quizzes.

Labs

- Labs on Monday from 1-4. You should plan on spending the entire time working with the help of the T/As.

Your T/As

- Kumar Abhishek: `kua3@lehigh.edu`
- Udom Janjarassuk: `udj2@lehigh.edu`
- Mustafa Kilinç: `mrk304@lehigh.edu`

- I am paying them to help out of my own pocket, so please think before you bug them

Grading

- 10% Participation
- 30% Labs and Problem Sets
- 30% Quizzes
- 30% Final Exam

Topics

1. Introduction to Algorithm Analysis
2. Sorting and Searching
3. Graph Algorithms
4. Numerical Algorithms

Course Objectives (IE170)

- Understand the basic principles of algorithm design, especially for applications in systems engineering;
- Understand basic techniques for analyzing the performance of algorithms;
- Develop an appreciation for the importance of implementing algorithms efficiently and the skills necessary for doing so;
- Develop an ability to solve systems engineering problems by designing and implementing an appropriate algorithm.
- Develop an ability to solve systems engineering problems by designing and implementing an appropriate algorithm.

Course Objectives

- Sharpen and extend basic computing and programming skills
- Learn various well-known sorting algorithms
- Develop deeper knowledge of advanced data structures and Java frameworks implementing these data structures
- Discover methods for representing discrete structures such as graphs and basic operations on graphs
- Learn and implement well-known algorithms for solving optimization problems on graphs
- Appreciate the importance of numerical methods through implementing core techniques such as solving linear systems and solving least squares problems

Great Expectations

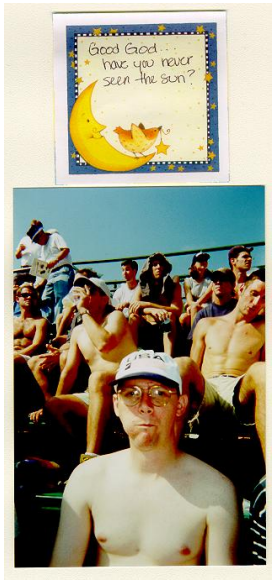
I am expected to...

- Teach
- Answer your questions
- Be at my office hours
- Give you feedback on how you are doing in a timely fashion

You are expected to...

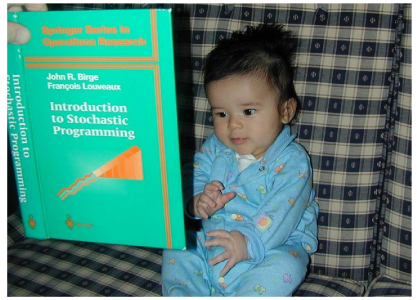
- Learn
- Attend lectures and participate
- Do the labs and problem sets
- Not be rude, if possible.
  - Sleeping
  - Talking
  - Interrupting
  - Cell Phones
  - Leaving in the middle of lecture

About me...



- B.S. (G.E.), UIUC, 1992.
- M.S., OR, GA Tech, 1994.
- Ph.D., Optimization, GA Tech, 1998
- 1998-2000 : MCS, ANL
- 2000-2002 : Axioma, Inc.
- Research Areas: Large Scale Optimization, High Performance Computing.
- Married. One child, Jacob, born 10/28/02. He is awesome.
- Hobbies: Golf, Integer Programming, Chess.

Picture Time



An Interesting Quote

Emeril—The Algorithm Maker

"Here is your book, the one your thousands of letter have asked us to publish. It has taken us years to do, checking and rechecking countless recipes to bring you only the best, only the interesting, only the perfect. Now we can say, without a shadow of a doubt, that every single one of them, if you follow the directions to the letter, will work for you exactly as it did for us, even if you have never cooked before."

–McCall's Cookbook (1963)

A Brief History of Algorithms

- According to the Oxford English Dictionary, the word algorithm is a combination of the Middle English word algorism with arithmetic.
- This word probably did not enter common usage in the English language until sometime last century.
- The word algorism derives from the name of an Arabic mathematician circa A.D. 825, whose surname was Al-Khwarizmi.
- Al-Khwarizmi wrote a book on solving equations from whose title the word algebra derives.

The First Algorithm

- It is commonly believed that the first algorithm was Euclid's Algorithm for finding the greatest common divisor of two integers, m and n, $(m \geq n)$.

Euclid's Algorithm(m, n)

1. Divide m by n and let r be the remainder.
2. If $r = 0$, then $\gcd(m, n) = n$.
3. Otherwise, $\gcd(m, n) = \gcd(n, r)$

Show Me the Algorithms

- Algorithms are literally everywhere you look.
- What are some common applications of algorithms?
  - Solving equations.
- Why is it important that algorithms execute quickly?

What is a Problem?

- Roughly, a problem specifies what set of outputs is desired for each given set of inputs.
- A problem instance is just a specific set of inputs.

Example: The Sorting Problem

- Input: A sequence of numbers $a_1, a_2, \ldots, a_n$
- Output: A reordering $a'_1, a'_2, \ldots, a'_n$ such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

What Do You Mean By "Solve"?

- Solving a problem instance consists of specifying a procedure for converting the inputs to an output of the desired form (called a solution).
- An algorithm that is guaranteed to result in a solution for every instance is said to be correct.
- In this class, we study (formally) how to prove that an algorithm is correct.
- And we also study how to formally characterize (analytically) the amount of effort required to produce a solution.

Aside: Pseudo-code

- The book will use pseudo-code to specify algorithms.
- The syntax will be similar to Java, but with departures for clarity.
- We will use descriptions in English to specify certain operations.
- The pseudo-code used in lecture may differ from that in the book. (I may use more Java-like syntax)
- Check the book for a detailed list of its pseudo-code conventions.

Another Example: Fibonacci Numbers

- Another simple example of the importance of efficient algorithms arises in the calculation of Fibonacci numbers.
- Fibonacci numbers arise in population genetics, as well as a host of other applications. (Including the analysis of Euclid's Algorithm!)
- The n^{th} Fibonacci number is defined recursively as follows:

$$\begin{aligned} F(0) &= 0, \\ F(1) &= 1, \\ F(n) &= F(n-1) + F(n-2) \ \forall n \in \mathbb{N}, n > 2. \end{aligned}$$

- How do we calculate $F(n)$ for a given $n \in \mathbb{N}$

Calculating Fibonacci Numbers

- Obvious Solution: A recursive function `Fib()`

```
Fib(int n)
{
   if (n == 0) {
     return 0;
   }
   else if (n == 1) {
      return 1;
   }
   return (Fib(n-1) + Fib(n-2));
}
```

Answering Some Questions

- How efficient is the recursive form of `Fib()`?
  - We will be able to answer this (analytically) by the end of the course.
  - We will answer it empirically in lab today.
- Is there a more efficient algorithm?
  - You will work on it in the lab.

What is a Data Structure?

- Computers operate on tables of numbers (the data).
- Within the context of solving a given problem, this data has structure.
- Data structures are schemes for storing and manipulating data that allow us to more easily see the structure of the data.
- Data structures allow us to perform certain operations on the data more easily.
- The data structure that is most appropriate depends on how the algorithm needs to manipulate the data.

Importance of Data Structures

- Specifying an algorithm completely includes specifying the data structures to be used (sometimes this is the hardest part).
- It is possible for the same basic algorithm to have several different implementations with different data structures.
- Which data structure is best depends on what operations have to be performed on the data.

Summing Up

- Algorithms that are both efficient and correct are a technology that must be developed.
  - (At least if you want a good grade in this class).
- Data structures allow us to represent relationships between various data and allow us to manipulate them effectively during an algorithm.
- Efficient algorithms enable us to solve important problems more quickly, which is critical for many applications.
- This is the focus of this class.

In Our Next Episodes...

- Read: CLRS Appendix A, B, and C.
- Read: CLRS Chapter 1, 2 and 10.
- Some basic sorting algorithms and introduction to analysis