Lecture 1: Introduction

CS 341: Algorithms

Tuesday, January 3\textsuperscript{rd} 2016
Outline For Today

1. Administrative Information
2. Overview of CS 341
3. Example 1: Sorting-Merge Sort-Divide & Conquer
Outline For Today

1. Administrative Information
2. Overview of CS 341
3. Example 1: Sorting-Merge Sort-Divide & Conquer
Instructor: Semih Salihoglu (semih.salihoglu@uwaterloo.ca)

TAs: Shayan Hassantabar, Woojung, Kim, Jian Li, Shikha Mahajan, Aayush Rajasekaran, Jose Serna, Dimitrios Skrepetos

Office Hours: Semih: TBA @ DC 3351

Website: https://www.student.cs.uwaterloo.ca/~cs341/

Piazza: (link will be on the course web site)

Textbook (Optional): Cormen, Leirerson, Rivest, Stein 3rd edition

Adjustments to the lectures of my sections:

- I am away for 3 lectures
  - Thursday March 16th: Yaoliang lecturing
  - March 21st & 23rd: Doug Stinson lecturing
Workload & Grade Distribution

- 5 Problem Sets: 25%
  - 2 or 3 of them will have small programming questions
  - Two or three weeks to complete each, on Fridays at noon
  - First one is out this Friday (Jan 6th). Due Jan 20th at noon.
  - Others: Jan 20, Feb 3, Feb 17, March 10th, March 31st.
- 1 Midterm: 25% (March 2nd, 7pm-8:50pm, STC 1012/AHS 1689)
- 1 Final: 50%: TBA

No late policy
Prerequisites

- **CS 240: Standard data structures**
  - Queues, stacks, heaps

- **Comfort with proofs**
  - Proof by induction
  - Proof by contradiction

- **Programming in a standard language:** TBA
Outline For Today

1. Administrative Information
2. Overview of CS 341
3. Example 1: Sorting-Merge Sort-Divide & Conquer
Why is CS 341 Important For You? (1)

- Algorithms is the heart of CS
- Appear in later courses
Connections to Other CS Courses

❖ CS 350: Operating Systems
  ➢ Scheduling Algorithms

❖ CS 482: Computational Biological Sequence Analysis
  ➢ Sequence Alignment Algorithms

❖ CS 485: Machine Learning
  ➢ Closest-pair/Clustering algorithms

❖ CS 456: Computer Networks
  ➢ Shortest-Paths Algorithms for Routing

❖ CO 331: Coding Theory
  ❖ Huffman’s Algorithm for Huffman Codes
Connections to Other Disciplines

◆ Biology
  - Sequence Alignment Algorithms

◆ Economics
  - Gale & Shapley’s Stable Marriage Algorithm
  - Shapley: Mathematician with a Nobel-prize in Economics

◆ Sociology: Milgram’s 6-degrees of separation phenomenon
  - Shortest paths algorithms
  - “The Small World Problem” Milgram, 1969
Why is CS 341 Important For You? (2)

- Algorithms is the heart of CS
- Appear in later courses
- Appear in technical interviews
  - *Willing to take bets on this!*
- For some of you, designing algorithms will be a lot of fun!
What is an Algorithm?

Informally: A well-defined procedure (or a set of instructions) to solve a computational problem?

◆ What’s a computational problem?
  ▪ Informally: Any problem w/ an input & an expected output

<table>
<thead>
<tr>
<th>Computational Problem</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sorting</td>
<td>An array of integers in arbitrary order</td>
<td>Same array of integers in increasing order</td>
</tr>
<tr>
<td>Matrix Multiplication</td>
<td>Two nxn matrices A, B</td>
<td>C=A*B</td>
</tr>
<tr>
<td>Traveling Salesman Problem</td>
<td>A set S of cities, and distances between</td>
<td>Minimum distance starting from city X</td>
</tr>
</tbody>
</table>
Example 1: Sorting

- **Input:** An array of integers in *arbitrary* order

  | 10 | 2  | 37 | 5  | 9  | 55 | 20 |

- **Output:** Same array of integers in *increasing* order

  | 2  | 5  | 9  | 10 | 20 | 37 | 55 |
Example 2: Matrix Multiplication

Input: 2 \( n \times n \) matrices \( A, B \)

\[
A = \begin{pmatrix}
2 & 1 & 5 \\
3 & 2 & 2 \\
1 & 4 & 6
\end{pmatrix}
\]

\[
B = \begin{pmatrix}
1 & 3 & 4 \\
2 & 1 & 1 \\
3 & 7 & 2
\end{pmatrix}
\]

Output: \( C = A \times B \)

\[
C = \begin{pmatrix}
19 & 41 & 18 \\
13 & 25 & 19 \\
27 & 49 & 20
\end{pmatrix}
\]
Example 3: Traveling Salesman Problem

- **Input:** Map of cities & distances between each pair of cities

- **Output:** \( \text{min distance it takes to go from a city } c_1, \text{ visit every other city once, and come back to } X? \)
  - Output is just an integer.

![Traveling Salesman Diagram]

- Cities: 1, 2, 3, 4, 5
- Distances and Travel Times:
  - 1 to 2: 5 hours
  - 1 to 3: 2 hours
  - 1 to 4: 9 hours
  - 1 to 5: 2 hours
  - 2 to 3: 3 hours
  - 2 to 4: 1 hour
  - 2 to 5: 4 hours
  - 3 to 4: 5 hours
  - 3 to 5: 1 hour
  - 4 to 5: 9 hours
Example 3: Traveling Salesman Problem

- **Input:** Map of cities & distances between each pair of cities

- **Output:** min distance it takes to go from a city $c_1$, visit every other city once, and come back to $c_1$?
  - Output is just an integer.

  Answer: 12
What is an Algorithm?

Informally: A well-defined procedure (or a set of instructions) to solve a computational problem.

Think of an algorithm as a: machine or software program
What is “Analysis” of Algorithms? (1)

Any machine/software program uses resources

Example Resources:
- Time (i.e., CPU time or number of operations)
- Memory (RAM) => referred more formally as “space”
- Network I/O or communication (ethernet)
What is “Analysis” of Algorithms? (2)

- Answering **how much** questions about the resources an algorithm uses:
  - How much time does it take to run Algorithm X?
  - How much memory does Algorithm X use?
  - How much network I/O does Algorithm X perform?

- CS 341: We’ll analyze time
  - Specifically: number of computer operations performed
Types of Algorithms

No real taxonomy; but 3 classic ways to classify algorithms

1. Serial vs Parallel
   - Serial: One operation at a time
   - Parallel: Multiple operations at a time

2. Deterministic vs Randomized
   - D: On multiple runs on same input, always do same ops
   - R: On multiple runs on same input, may do different ops

3. Exact vs Approximate
   - Exact: Exact output
   - Approximate: Approximate output

CS 341: serial, deterministic, exact algorithms
Fundamental (& Fast) Algorithms to Tractable Problems

- MergeSort
- Strassen’s MM
- BFS/DFS
- Dijkstra’s SSSP
- Kosaraju’s SCC
- Kruskal’s MST
- Floyd Warshall APSP
- Topological Sort
- ...

Common Algorithm Design Paradigms

- Divide-and-Conquer
- Greedy
- Dynamic Programming

Mathematical Tools to Analyze Algorithms

- Big-oh notation
- Recursion Tree
- Master method
- Substitution method
- Exchange Arguments
- Greedy-stays-ahead Arguments

Intractable Problems

- P vs NP
- Poly-time Reductions
- Undecidability

Other (Last Lecture)

- Randomized/Online/Parallel Algorithms
Before/After CS 341

1. Fundamental Algorithms
2. Fundamental Algorithm Design Paradigms
3. Tractability/Intractability

Will also learn about some CS history.
### A Comment About Tractability/Intractability

<table>
<thead>
<tr>
<th>Computational Problem</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sorting</strong></td>
<td>An array of integers in arbitrary order</td>
<td>Same array of integers in increasing order</td>
</tr>
<tr>
<td><strong>Matrix Multiplication</strong></td>
<td>Two nxn matrices A, B</td>
<td>C=A*B</td>
</tr>
<tr>
<td><strong>Traveling Salesman Problem</strong></td>
<td>A set S of cities, and distances between each pair of cities</td>
<td>Minimum distance starting from city X, visiting each city once and come back to X</td>
</tr>
</tbody>
</table>

- **Tractable**
- **Intractable**