Lecture 1: Introduction

CS 341: Algorithms

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

Administrative Info (1)

- Instructor: Semih Salihoglu (semih.salihoglu@uwaterloo.ca)
- TAs: Many, please see the website
- Office Hours: Semih: Thursdays at 4pm @ DC 3351
- Website: https://www.student.cs.uwaterloo.ca/~cs341/
- Piazza: https://piazza.com/uwaterloo.ca/winter2019/cs341/home
- Weekly Tutorials: 2 sections on Fridays. Website for time/location.
- Textbook: Cormen, Leirserson, Rivest, Stein 3rd edition
  - Online version available once you login to library.uwaterloo.ca
    - https://tinyurl.com/yc7bhe3y (if doesn't work, do a keyword search)

Administrative Info (2)

- Workload & Grade Distribution
  - 5 Problem Sets: 30%
    - 2 or 3 of them will have programming questions
    - Two or three weeks to complete each, on Fridays at 6pm
    - First one is out this Friday. ***Due Jan 25th***
    - Others: Check the website
  - 1 Midterm: 25%, Feb 26th, 7pm-8:50pm, TBA
  - 1 Final: 45%: 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

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

Example 1: Sorting

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

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

Example 2: Matrix Multiplication

- Input: 2 nxn matrices A, B

2	1	5
3	2	2
1	4	6

A

1	3	4
2	1	1
3	7	2

B

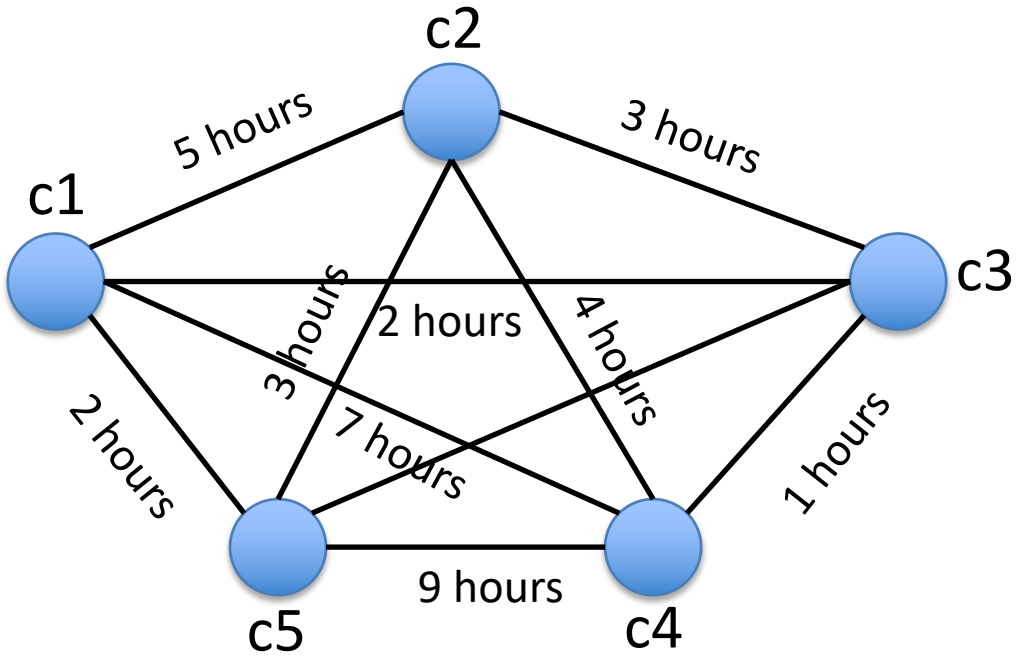
- Output: C=A*B

19	41	18
13	25	19
27	49	20

C

Example 3: Traveling Salesman Problem

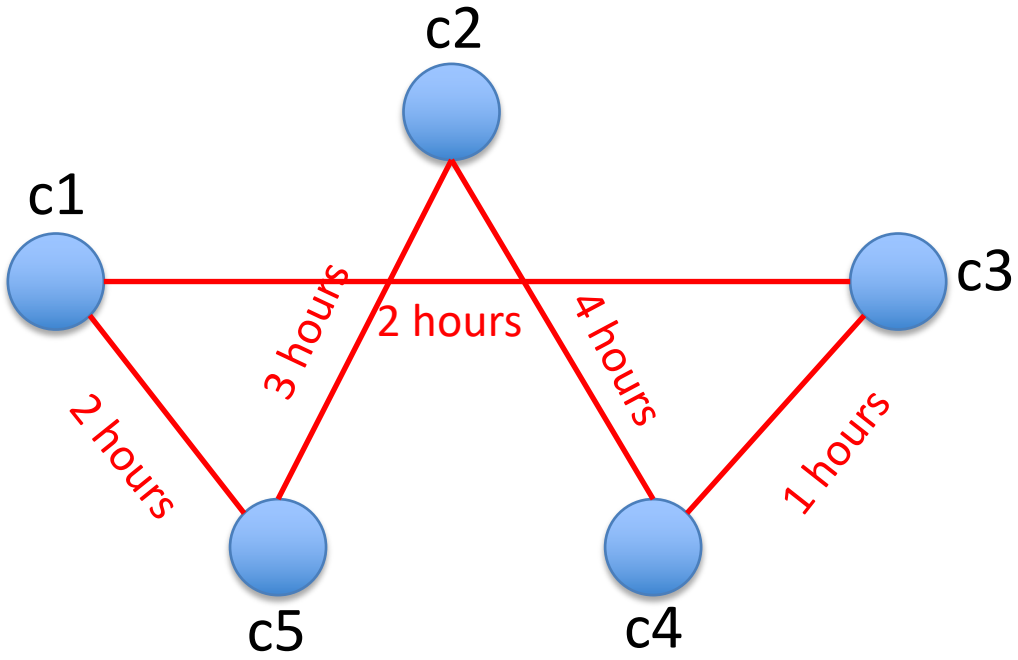
- Input: Map of cities & distances between each pair of cities



- Output: min distance it takes to go from a city c1, visit every other city once, and come back to X?
  - Output is just an integer.

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 c1, visit every other city once, and come back to c1?
  - 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 ← one or two algorithms towards the end
     - Exact: Exact output
     - Approximate: Approximate output
- CS 341: serial, deterministic, exact algorithms

CS 341 Diagram

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

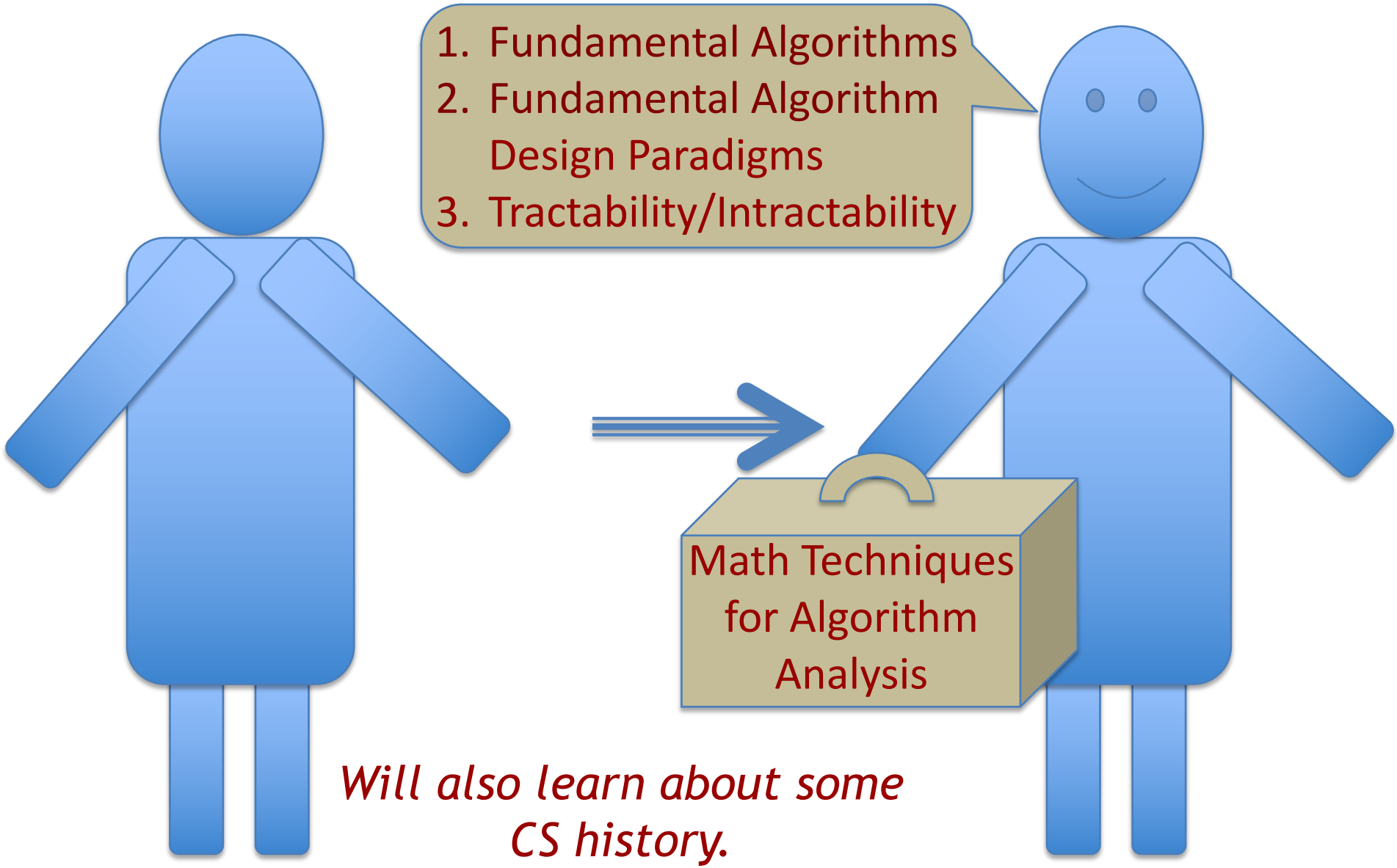
Intractable Problems

- P vs NP
- Poly-time Reductions
- Undecidability

Other (Last Lecture)

- Randomized/Online/Parallel Algorithms

Before/After CS 341



Will also learn about some CS history.

A Comment About Tractability/Intractability

Tractable

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

Intractable