L28: LDOC and the Limits of Computing

Alex Steiger

CompSci 201: Spring 2024

4/24/2024

1

Logistics, coming up

- Extra credit! 3 surveys for 0.5% final grade each:
  - Official course evals (>70% completion)
  - End-of-semester survey (individual completion)
  - AiiCE survey (>70% completion)
  - **Due 4/27 @ midnight**
- Next week on Tuesday, 4/30
  - Final exam, 9am-12pm
  - Required, comprehensive

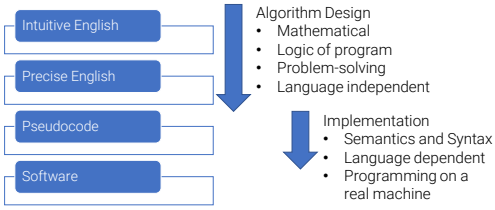
2

Looking back at our semester

3

What are algorithms?

Loosely speaking: A precise sequence of unambiguous steps that effectively compute an output given an input.



4

What is code?

In order to execute an algorithm on a real computer, we must write the algorithm in a formal language. An algorithm so written is a **program**.

In this class we explore both:

Theory
- Design an algorithm
- Analyze performance
- Data structure tradeoffs

Practice
- Write a Java program
- Debug/test
- Measure performance

5

Why does efficiency matter?

- You wrote the next big social media app:
  - Will it work if it has 1 billion users?
  - What about on a phone with limited memory?
- In the sciences, discovery depends on computing with big data:
  - Sequencing the human genome
  - Surveying millions of images in astronomy
  - Processing data logs from the CERN collider
- Pushing the limits of current technology:
  - Virtual / augmented reality?
  - Deep neural networks for large scale machine learning?

6

Some specifics you ~~will~~ did learn

Data Structures
- Arrays
- Lists: ArrayList and LinkedList
- Sets: HashSet and TreeSet
- Maps: HashMap and TreeMap
- Stacks, Queues, Priority Queues / Heaps
- Trees: Binary Search Trees
- Graph representations

Algorithms
- Iterative
- Hashing
- Big O Asymptotic Analysis
- Recursive
- Sorting
- Greedy
- Graph

Software
- Java API
- Objects, Classes
- Interfaces, implementations
- Testing, Debugging

7

Informal goals for the course

- Make or deepen a friendship with someone else passionate about computer science.
- Develop a new appreciation of computing phenomena you see in the real world.
- Experience joy when your program *works*, even if it took a while to get it there.
- WOTO: WOrking TOgether
- Stay safe and healthy, physically and mentally

8

Who to Thank

- Violet (Teaching Associate), Mark and Eamon (Grad TAs) working behind the scenes to make this work at scale
- ***All of our undergrad TAs***! Providing feedback, helper hours, running discussions, etc.
- Your fellow students! Discussion groups, friends, project partners, etc.

9

What I'm thankful for

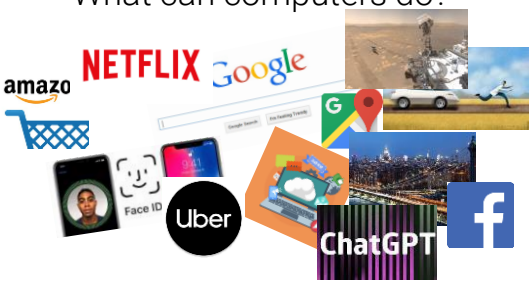
- Safety to gather and be together
- My teaching team
- All of you (why am I here?!?!?)

10

Parting Thoughts: What computers can and can't do?

12

What can computers do?



13

What can't computers do?

- Some problems ***cannot be solved at all***
  - One program detects all infinite loops
- Some problems ***cannot be solved efficiently***
  - Listing all N-bit sequences of 0's and 1's
- Some problems can be ***approximately solved***
  - AI, ML, close-to-optimal is good enough

14

Halting Problem

- Can we write doesHalt as specified? *Suppose so!*
  - Like the Java Compiler: reads a program

```
public class ProgramUtils
    /**
     * Returns true if progname halts on input,
     * otherwise returns false (infinite loop)
     */
    public static boolean doesHalt(String progname){
    }
}
```


15

Can we confuse doesHalt?

- What if **`doesHalt(confuse)`** returns true?
  - Then **`confuse()`** does not halt (see below)
- What if **`doesHalt(confuse)`** returns false?
  - Then **`confuse()`** does halt (see below)

```
public static boolean confuse(){
    if (ProgramUtils.doesHalt(confuse)) {
        while (true) {
            // do nothing forever
        }
    }
}
```


16

Formal proof is a bit more challenging…

- Alan Turing first showed this for programs: 1936
  - Had to formally specify what a program was
  - Needed to invent concept of Turing Machine
  - Also demonstrated by Alonzo Church
- Cantor showed # Real Numbers > # Rationals
  - So-called diagonalization, 1891
  - Ridiculed by establishment
  - Argument essential to above

17

Shortest/Longest Path; P and NP

- Dijkstra's Algorithm one example
  - Others: Floyd-Warshall and more
  - Very efficient graph algorithms,
- Longest Path? No efficient solution known
  - Easy to verify "is this path greater than length k"
  - Exponentially many paths

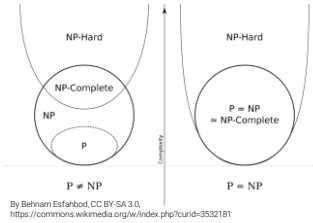
18

P vs NP

- P is the set of (algorithmic) problems that can be ***solved*** in time that is polynomial in the size of the input (polynomial time).
  - i.e., can solve with a program that is O(1), O(N), O(Nlog(N)), $O(N^2)$, $O(N^3)$, …, $O(N^{128})$, …
- NP is (roughly) the set of (algorithmic) problems for which a solution can be ***verified*** in polynomial time.

19

P ?= NP



By Behnam Esfahbod, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=3532181

- Most think P != NP
- Greatest outstanding question in theoretical computer science
- Proof is worth a $1M prize from the Clay Mathematics Institute

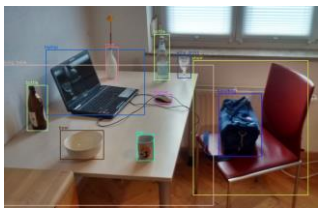
20

"Easy" Hard Problems

- Some problems are hard to solve but easy to approximate:
  - Can't write a program to give you the optimal solution efficiently but can find something within ϵ of optimal in polynomial time.
  - Greedy, randomized, etc.
- Some problems are hard to prove things in theory but easy to solve in practice
  - Can't prove much but it works well in practice

21

AI/ML often work with experimental algorithms for hard problems

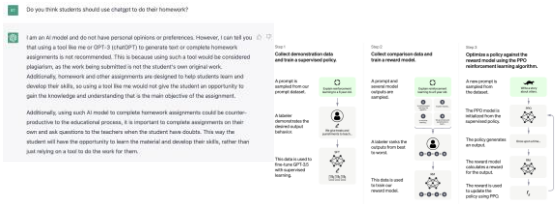


Common idea: Use a computer to learn a function/neural network that approximates a large dataset.

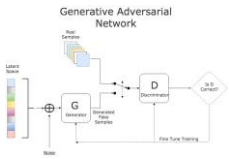
- Image segmentation/ classification
- Face/speech recognition
- Machine translation
- Text generation
- Reinforcement learning
- Robotics
- …

22

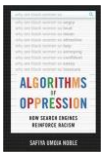
Artificial Intelligence: ChatGPT and Reinforcement Learning



More Concerning? Deepfakes and Generative Adversarial Networks (GANs)



What should computers do?



We need citizens computationally equipped to deal with these algorithmic systems in theory and in practice

26

Who has gone before you? People in CS

27

What will you do?

- Not everyone wants to be a software engineer
  - Diplomat, lawyer, physician, entrepreneur,
  - Musician, teacher, data scientist, …
- Not all jobs at tech companies are SWE
  - UI, UX, PM, …
- Some non-tech companies have tech jobs
  - Healthcare? Aerospace? Biotech? Finance? Non-profit? NASA?
- Grad school? Research? Teaching?

28

What I'm thankful for

- Safety to gather and be together
- My teaching team
- All of you (why am I here?!?!?)