Lecture 1

Administrative

A Section:
Beginning - Mar 16: Patrick
Mar 18 - End: Chandra
Chat Moderator: Vasilis

B Section:
Beginning - Mar 23: Patrick
Mar 25 - End: Chandra
Chat Moderator: Vasilis

A vs B: different lecturers
Same Lab/HW/Exams
Groups across sections etc!

Labs: Start today (tomorrow now)
HW 0/Quiz 0: available soon

Quiz: due Mon 10a
HW: due Wed 10a
Exams: synchronous, proctored via Zoom
MT1: Mar 1 6:30p (next day early morning)
MT2: Apr 12 6:30p

DRES: Send LOA to Patrick & Chandra well before Midterm

Webpage:
Lecture Schedule/Readings/Recordings
Labs, HW
Office Hours Schedule/Procedures
(Too Many) Policies

Other Sites:
Piazza: Q&A + Announcements
Discord: OH + Socialization
Gradescope: HW/Exam grading
PrairieLearn: Quizzes/Additional Practice

Course Structure/Philosophy
A lot of material very quickly
refresh prereqs CS 173/225
→ HWO/Q420

What are we talking about?

- Computation, formally (automata, TMs)
- How to compute “efficiently” (algorithms design)
- Limits of computation (undecidability & NP-Hardness)

Key Idea: Reduction $A \leq B$

Informally: problem $A$, convert it to problem $B$
apply known solution for $B$ to solve $A$.

Why?

- Computers existed for millennia
  - many physical computers share
    - “same” limitations
  - capture this into abstract “idealized” mathematical
    machine
  - physical world (model of computation)
    implement

- efficiency is important & difficult
  - e.g. multiplying $2$ n-digit #s:
    \[
    \begin{array}{c}
    3141 \\
    \times 2718 \\
    \hline
    25128 \\
    3141 \\
    \hline
    314107
    \end{array}
    \]
  - “alternatives”
    - e.g. lattice multiplication
\[
\begin{align*}
&\frac{21987}{6282} = 3.418537238 \\
&O(n^2) \\
&\text{Kolmogorov conjectured } \Omega(n^2) \\
&\text{Karatsuba, 1960 } O(n^{1.58}) \\
&\text{Schönhage-Strassen, 1971 } O(n \log n \log \log n) \\
&\text{Furer, 2008 } O(n \log n 2^{O(\log^* n)}) \\
&\text{Harvey - van der Hoeven, 2019 } O(n \log n) \\
\end{align*}
\]

Limit of computation also useful & surprising.

Post Correspondence Problem:

Can I arrange copies of these tiles so the top matches the bottom?

\[
\begin{array}{c|c|c}
abb & ba & a \\
bbb & abb & b \\
\end{array}
\]

Cannot be solved w/ a computer???

How?

Common to many "computers" throughout history to store information as "strings".

Formally: alphabet \( \Sigma \) finite set.
Formally: alphabet $\Sigma$ is finite set

- $\Sigma = \{0, 1\}$
- $\Sigma = \{a, \ldots, z\}$
- $\Sigma = \{\langle A440\rangle, \langle C261.63\rangle, \ldots\}$

String is a finite sequence of symbols from $\Sigma$.

- $01001111$ 'string'
- $\langle C261.63\rangle \langle F#369.99\rangle \langle G392.00\rangle$

Simplistically talk about "Decision" problem
answer is Y/N.

- Post Correspondence Problem
- Given a list, is it sorted?

Language: a set of strings not necessarily finite.

$L = \{s \mid \text{the answer in problem is "yes"}$

$P: \text{Given } x, \text{ is } x \in L?$

deciding Y/N problems = solving "language membership"

Defining string recursively:

a string is either

- $\varepsilon$ (the empty string)
- $a \cdot x$ $a \in \Sigma, x$ string

String = $s \cdot t$ string = $s \cdot (t \cdot \text{string}) = \ldots = s \cdot (t \cdot ((t \cdot \text{string}) \ldots \text{string}))$

operations on strings

e.g. length $|\text{string}| = 6$. 
**Operations on strings**

- **Example:** \( |\text{length of string}| = 6 \).
  
  \( |w| = \# \) symbols in \( w \)

- \( |w| = \begin{cases} \ 0 & \text{if } w = \epsilon \\ \ |1| + |x| & \text{if } w = ax \end{cases} \)

- **Example:** concatenation \( x \)
  
  \( \text{HEAD} \cdot \text{ACHE} = \text{HEADACHE} \)

- \( w \cdot z = \begin{cases} \ z & w = \epsilon \\ a(x \cdot z) & w = ax \end{cases} \)

**Thm.** \( |w \cdot z| = |w| + |z| \)

**Pf.** Let \( w, z \) be arbitrary strings

**IH:** Assume \( |x \cdot z| = |x| + |z| \) for all \( x \)

where \( |x| < |w| \)

There are two cases

- \( w = \epsilon \) \( \rightarrow \) \( |w \cdot z| = |\epsilon \cdot z| \)
  
  \( = |z| \) \hspace{1cm} \text{def of } \epsilon

  \( = 0 + |z| \) \hspace{1cm} \text{add 0}

  \( = |z| + |z| \) \hspace{1cm} \text{def of } 1

  \( = |w| + |z| \) \hspace{1cm} w = \epsilon

- \( w = ax \) \( \rightarrow \) \( |w \cdot z| = |ax \cdot z| \)

  \( = |a(x \cdot z)| \) \hspace{1cm} \text{def of } a

  \( = 1 + |x \cdot z| \) \hspace{1cm} \text{def of } 11

  \( = 1 + |x| + |z| \) \hspace{1cm} IH

  \( = |ax| + |z| \) \hspace{1cm} \text{def of } 11

  \( = |w| + |z| \) \hspace{1cm} w = ax \)
Ops on Languages

Given langs \( L, M \)

\[ L \cdot M = \{ x \cdot y | x \in L, y \in M \} \]

\[ \Sigma^0 = \{ \varepsilon \} \]

\[ \Sigma^1 = \{ a \varepsilon | a \in \Sigma \} \]

\[
\text{for } n \geq 1, \quad \Sigma^n = \sum \sum \Sigma^{n-1} \]

is the set of length \( n \) strings

\[ \Sigma^* = \bigcup_{n \geq 0} \Sigma^n \]

is the set of all possible strings

\[ \Sigma^* = \{ \varepsilon, a \varepsilon, aa \varepsilon, \ldots \} \]

is a countably infinite set

\[ |\Sigma^*| = |\mathbb{N}| \]

\( L \subseteq \Sigma^* \) for all langs \( L \)

Every program is a string

\# programs = |\Sigma^*|

\# decision problems = \# languages = |\mathcal{P}(\Sigma^*)| \neq |\Sigma^*|

Exist (uncountably many) problems that cannot be solved viz a computer.