CS 121: Lecture 6
Code = Data

Madhu Sudan

`https://madhu.seas.Harvard.edu/courses/Fall2020`

Book: `https://introtcs.org`

How to contact us:
- The whole staff (faster response): CS 121 Piazza
- Only the course heads (slower): cs121.fall2020.course.heads@gmail.com

Where we are:

Part I: Circuits:
Finite computation, quantitative study

Part II: Automata:
Infinite restricted computation, quantitative study

Part III: Turing Machines:
Infinite computation, qualitative study

Part IV: Efficient Computation:
Infinite computation, quantitative study

Part V: Randomized computation:
Extending studies to non-classical algorithms

- Definition of Circuits
- Universality of NAND
- All functions can be computed
- $\forall f : \{0,1\}^n \rightarrow \{0,1\}$: Size(f) $\leq O\left(\frac{2^n}{n}\right)$
- Claimed: $\exists f, \backslash\mathrm{Size}(f) \geq \Omega\left(\frac{2^n}{n}\right)$
- Today: Will prove above.
- Show: Code = Data
- Show how to interpret Data as Code.

Today: Code as Data

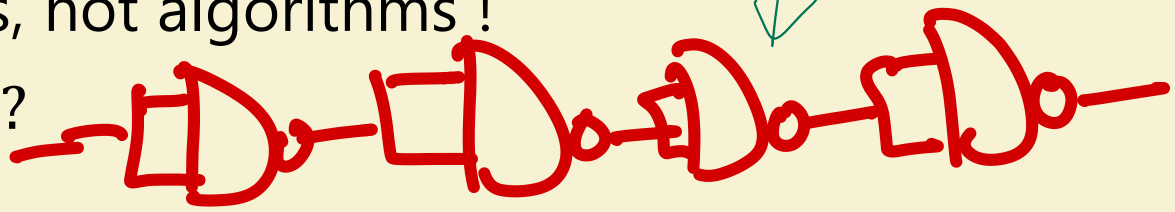
- Circuits can be represented by bits
- Exercise break: Quantify above. Prove lower bound on Circuit size (for hardest function).
- Universality: Circuit Interpreter $I(C,x) = C(x)$
  - As immediate consequence of “Code as data” - Inefficient
- Efficient Circuit Interpreter (sketch)
- Exercise break: Some Ingredients

Notation: $\mathbf{SIZE}(s)$

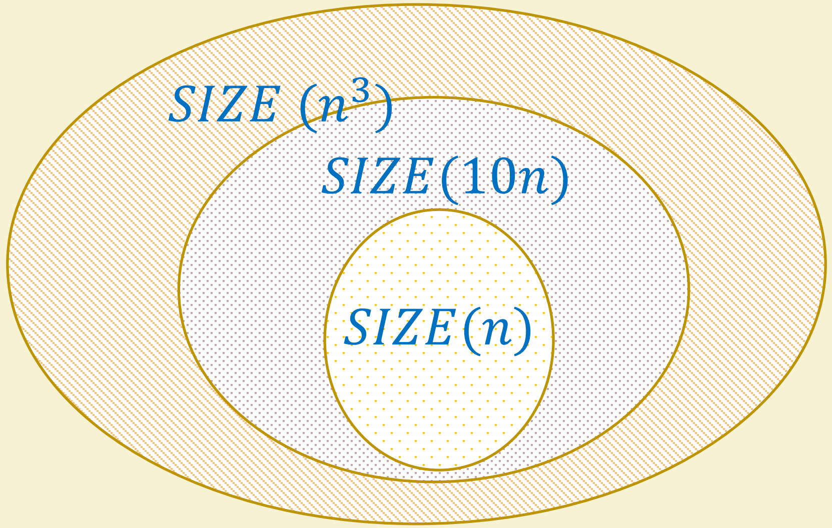
Answer was: The circuit is not in SIZE(anything) since circuits are not functions.

Underlying function is in SIZE(1).

- $\text{SIZE}(s) = \{f:\{0,1\}^n \to \{0,1\} \mid \exists C$ with $\le s$ NAND gates computing $f\}$
- $\text{SIZE}(s)$ = Our first complexity class!
  - Always a set (aka "class") of functions, not algorithms !
  - Is the following in $\text{SIZE}(3)$? $\text{SIZE}(10)$?



- $\text{ALL}_n = \{f:\{0,1\}^n \to \{0,1\}\}$
- Thm: $\text{ALL}_n \subseteq \text{SIZE}\left(O\left(\frac{2^n}{n}\right)\right)$
- (Claimed) Thm: $\text{ALL}_n \not\subseteq \text{SIZE}\left(o\left(\frac{2^n}{n}\right)\right)$



Reminder: Circuit ≡ Straightline Program

Temp[0] ← NAND($X[0], X[1]$)

Temp[1] ← NAND($X[2], X[2]$)

…

Temp[i] ← NAND(Temp[j], $X[k]$)

…

Y[0] ← NAND($X[0], X[1]$)

…

Y[m-1] ← NAND($X[0], X[1]$)

Encode to $\{0,1\}^*$ in class

(0 ← X0, X1)

(1 ← X2, X2)

⋮

(i ← Tj, Xk)

i, j, k … ∈ [n + s]

s = # lines.

Exercise Break 1:

1. Make Representation quantitative:
   - Give (prefix-free) E: Circuits $\to \{0,1\}^*$, such that $\forall\, C$ with $\leq s$ gates, $|\boldsymbol{E(C)}| = \boldsymbol{O(s \log s)}$
2. Show $|\boldsymbol{SIZE(s)}| = \boldsymbol{2^{O(s \log s)}}$ $\qquad 2^{O(s\log s)}$
3. Show $\exists f: \{0,1\}^n \to \{0,1\}$ s.t. $f \notin \mathrm{SIZE}\left(o\left(\frac{2^n}{n}\right)\right)$ $\left(\Leftrightarrow \ \mathbf{ALL}_{\boldsymbol{n}} \not\subseteq \boldsymbol{SIZE}\left(\boldsymbol{o}\left(\frac{\mathbf{2^n}}{\boldsymbol{n}}\right)\right)\right)$

Bonus question (0 points):
Why do Boaz Barak's slides associate this picture with 3rd exercise!

Representation

S gates: Each gate = 3 variables

– represent each line by $3\log(\cancel{3}S) + O(1)$

– total $S \times 3\log(\cancel{3}S) + O(1)$

Prefix-free $\rightarrow O(\log S) + \quad O(S\log S)$

Part 1: Representation:

- We represent the circuit as a list of lines.
- So need a prefix-free representation of line.
- line = ordered tuple of 3 variables; Each variable from a set of $n+s$ variables $\leq 3s$ variables

n inputs

s temp variables

[Can use $n \leq 2s$, since other inputs not involved in any gate]

$\Rightarrow$ line requires $3\log(3s)$ bits.

$\Rightarrow$ Circuit requires $s \times$ (bits/per line) $= s \times 3\log 3s = O(s\log s)$

$\boxtimes$

Part (2):

- Let $CIRCUIT(s) = \{$all circuits with at most s NAND gates$\}$
- We will prove $|SIZE(s)| \leq |CIRCUIT(s)| \leq |\{0,1\}^{O(s\log s)}|$

$$= 2^{O(s\log s)}$$

- Consider the map $A : SIZE(s) \to CIRCUIT(s)$

where given $f \in SIZE(s)$, $A(f)$ is a circuit of size $\leq s$ that computes f.

A is 1-1 since $A(f) = A(g) \Rightarrow f = g$ [any given circuit computes one function]

$\Rightarrow |SIZE(s)| \leq |CIRCUIT(s)|$

- From Part 1 of Exercise we have

$$E : \mathrm{CIRCUIT}(s) \longrightarrow \{0,1\}^{O(s\log s)}$$ that is 1-1.

$$\Rightarrow \quad |\mathrm{CIRCUIT}(s)| \leq \left|\{0,1\}^{O(s\log s)}\right| = 2^{O(s\log s)}$$

$\boxtimes$

<u>For part (3)</u>, recall $ALL_n = \{ f : \{0,1\}^n \to \{0,1\} \}$

$$\mathrm{SIZE}\left(o\left(\frac{2^n}{n}\right)\right) = \left\{ f \in ALL_n \;\middle|\; f \text{ has circuits of size } o\left(\frac{2^n}{n}\right) \right\}$$

- We wish to show

$$ALL_n \not\subseteq \mathrm{SIZE}\left(o\left(\frac{2^n}{n}\right)\right)$$ [$\exists f \in ALL_n$ that does not have $o\left(\frac{2^n}{n}\right)$ sized circuits]

- BIG BODY SMALL SHIRT Principle

if $|A| > |B|$ then $A \nsubseteq B$

- Apply to $A = ALL_n \Rightarrow |A| = 2^{2^n}$ [can you prove this?]

$B = SIZE\left(o\left(\frac{2^n}{n}\right)\right) \Rightarrow |B| = 2^{O\left(o\left(\frac{2^n}{n}\right) \cdot \log \frac{2^n}{n}\right)}$

$= 2^{o\left(\frac{2^n}{n} \cdot n\right)} = 2^{o(2^n)}$

So. $|A| > |B| \Rightarrow A \nsubseteq B$

$\Rightarrow ALL_n \nsubseteq SIZE\left(o\left(\frac{2^n}{n}\right)\right)$

$\boxtimes$

Interpreting Code

- Objective: Show Data representing Code can be interpreted as code.
- Have just shown: $\exists E$: Circuits $\mapsto$ binary string, 1-to-1

$$C \text{ has } \leq s \text{ NAND Gates} \quad \Rightarrow \quad |E(C)| = O(s \log s)$$

- EVAL: $(E(C), x) \mapsto C(x)$, $\forall C$ with n inputs, $x \in \{0,1\}^n$
  - EVAL is a partial function – why?

$$\mathrm{EVAL}_{m,n} : \{0,1\}^{m+n} \cancel{\times \{0,1\}^n} \to \{0,1\}$$

- $\mathrm{EVAL}_{m,n}$ = restriction of EVAL to $E(C) \in \{0,1\}^m, x \in \{0,1\}^n$
  - Thm: $\mathrm{EVAL}_{m,n}$ computed by circuit of size $O(2^{m+n})$
  - Proof: Obvious!
  - Implication: Power of Code↔Data-duality!

REST OF THE LECTURE:

(1) You should know the theorem being proved (Theorem 5.10 in text) but proof is not necessary (for hw, tests...) (you can read the proof if interested!)

(2) You should read the "Extended Turing Church Thesis (circuit version)" & be aware of the statement + implications.

Interpreting Circuits Efficiently.

m+n vs. m

- ignore the difference
- there are only 2m real inputs

- Goal: Show $\mathrm{EVAL}_{m,n} \in \mathrm{SIZE}\left(O\left((m+n)^2 \log^2 n\right)\right)$
  - Theorem 5.3 in Barak's IntroTCS.
  - Best bound in literature: close to $O\left((m+n)\log^2(m+n)\right)$
  - Great, but not "Meta-circular interpreter" (small interpreter that interprets bigger functions).

(1) length of input = n

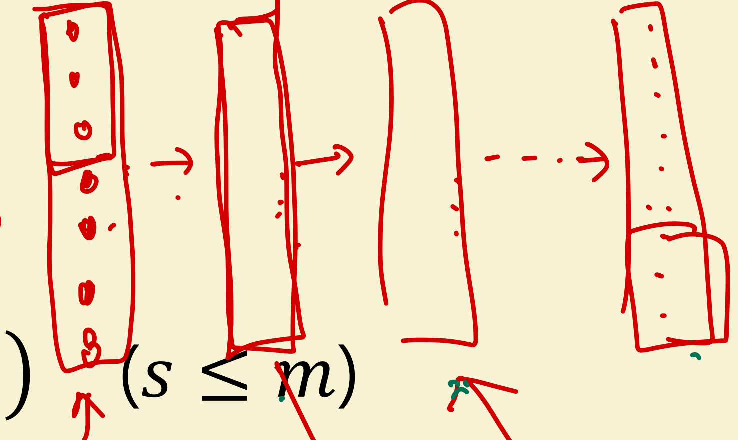
(2) length of circuit = S

(3) Encoding leng = m ≥ S

Sketch of **EVAL**

n

$k \leftarrow \mathrm{NAND}(i,j)$



- Recall: $E(C) = ((i_0, j_0, k_0) \ldots (i_{s-1}, j_{s-1}, k_{s-1}))$ $(s \le m)$
- Define: $W_t \in \{0,1\}^{n+s}$: Values of n inputs, s TEMPs after t execution steps

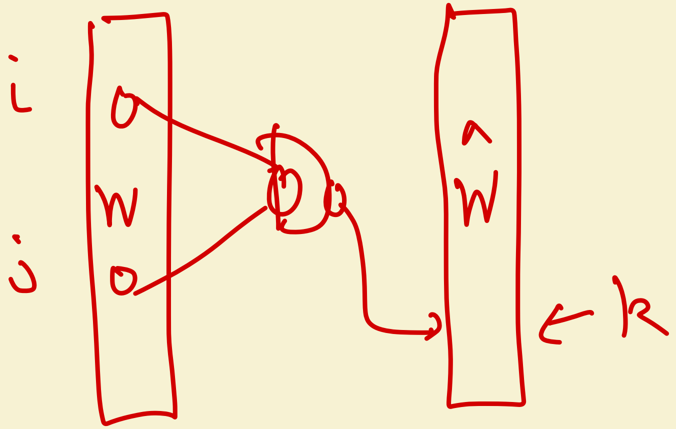
initially

$W_t \to W_{t+1}$

- Define:
  - $\mathrm{EVAL-ITER}: (E(C), x, t) \mapsto W_t$
  - $\mathrm{EVALHELP}: (W_{t-1}, i_t, j_t, k_t) \mapsto W_t$;
  - $\mathrm{EVAL-ITER}(E(C), x, t) = \mathrm{EVALHELP}(\mathrm{EVAL-ITER}(E(C), x, t-1), i_t, j_t, k_t)$
  - Suffices to show $\mathrm{EVALHELP} \in \mathrm{SIZE}((m+n)\log m + n)$

$\mathrm{Temp}(k_t) \leftarrow \mathrm{NAND}(\mathrm{Temp}(i_t), \mathrm{Temp}(j_t))$

Sketch of **EVALHELP**



- Key Ingredients:
  - $\mathrm{LOOKUP}(W,i) = W_i$ where $W = W_0 \ldots W_{m-1} \in \{0,1\}^m$, $i \in [m]$ represented in binary.
  - $\mathrm{UPDATE}(W,k,b) = \widehat{W}$ where $\widehat{W}_k = b$ and $\widehat{W}_\ell = W_\ell$ for $\ell \neq k$
  - Claims:
    - $\mathrm{LOOKUP} \in \mathrm{SIZE}(m)$
    - Exercise: $\mathrm{UPDATE} \in \mathrm{SIZE}(m^2)$ (even better $\mathrm{SIZE}(m \log m)$)
      - Don't have to work out details. Think of the high-level plan.
  - $\mathrm{EVALHELP}(W,i,j,k) = \mathrm{UPDATE}\Big(W,k,\mathrm{NAND}\big(\mathrm{LOOKUP}(W,i),\mathrm{LOOKUP}(W,j)\big)\Big)$

Exercise Break 2:

- $\mathrm{UPDATE}(W, k, b) = \widehat{W}$ where $\widehat{W}_k = b$ and $\widehat{W}_\ell = W_\ell$ for $\ell \neq k$

  $W, \widehat{W} \in \{0,1\}^m, k \in [m]$ represented in binary

- Exercise:
  - Show $\mathrm{UPDATE} \in \mathrm{SIZE}(m^2)$ (even better $\mathrm{SIZE}(m \log m)$)
    - Don't have to work out details. Think of the high-level plan.

Note that $\mathrm{UPDATE}(W,k,b)_i = b$ if $k = i$

$= W_i$ if $k \neq i$

Write $k = k_0 \ldots k_\ell$ where $\ell = \log(m+n)$

so that. $k = k_0 + 2k_1 + \ldots 2^\ell k_\ell$.

similarly $i = i_0 \ldots i_\ell$

let

then $\mathrm{UPDATE}(W,k,b)_i = \mathrm{IF}\left(\delta_i(k_0 \ldots k_\ell),\ b,\ W_i\right)$

where $\delta_i(k_0 \ldots k_\ell) = \overset{\ell}{\underset{j=0}{\mathrm{AND}}} \left[\mathrm{NOT}(\mathrm{XOR}(i_j, k_j))\right.$

- δ_i requires $O(\ell)$ gates;

$\mathrm{UPDATE}(w,k,b)_i$ also requires $O(\ell)$ gates

$\Rightarrow$ UPDATE requires $O(m\ell) = O(m \log m)$ gates

Circuits: What you need to know

Theorem I: Every function $f:\{0,1\}^n \to \{0,1\}$ can be computed by circuit of size $O(2^n/n)$.

Theorem II: Some functions $f:\{0,1\}^n \to \{0,1\}$ cannot be computed by circuits of size $o(2^n/n)$.

SIZE Hierarchy Theorem: Book + Section/HW

Thm 5.11: $\exists C$ ($C = 1000$ will do) .s.t $\forall s < \frac{2^n}{Cn}$, $SIZE_{n,1}(s) \subsetneq SIZE_{n,1}(C \cdot s)$

* If f outputs m bits then add factor m to Thm I,II

Extended Church Turing Thesis (circuit version)

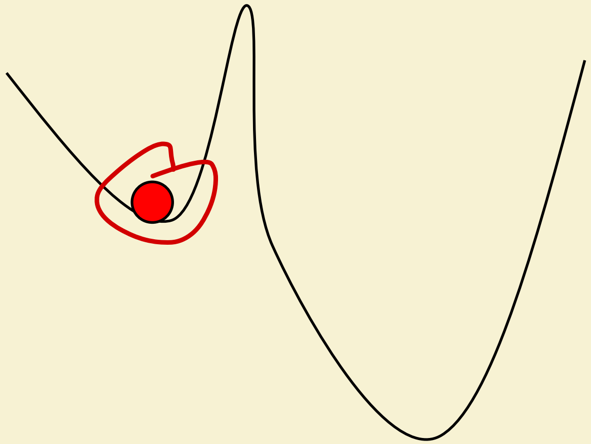
If $f:\{0,1\}^n \to \{0,1\}^m$ can be computed in the physical world using s resources then f can be computed by circuit of $\approx s$ (e.g. $O(s^2)$ or $O(s^3)$) gates.

(finite function version – we'll see unbounded function version soon)

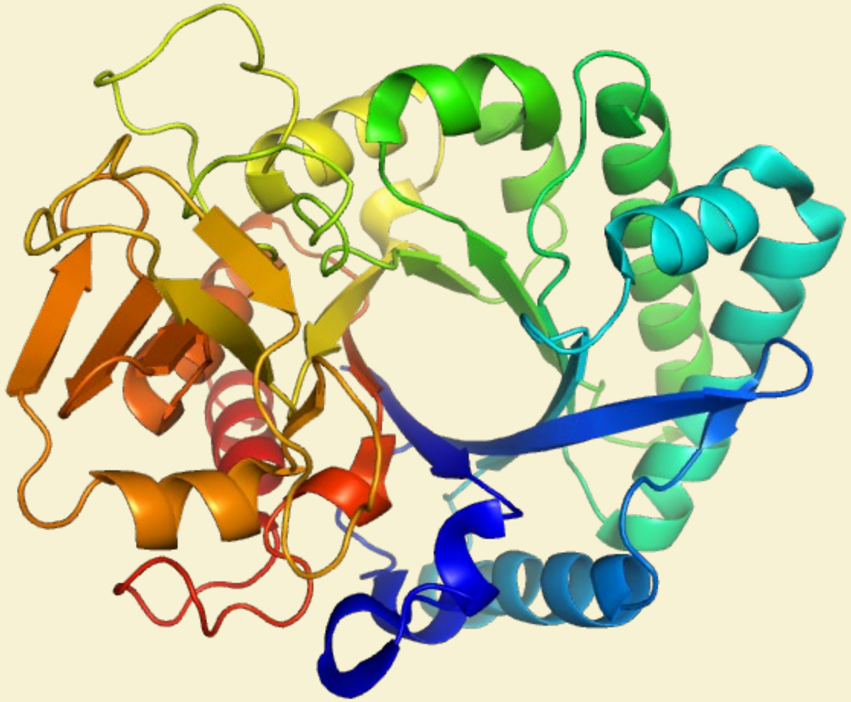
TL;DR: So far still stands. Only serious challenge is *quantum computing* ~~which we'll see later.~~

Non-serious challenges: *(Following slides stolen from Boaz Barak who stole it from Scott Aaronson)*

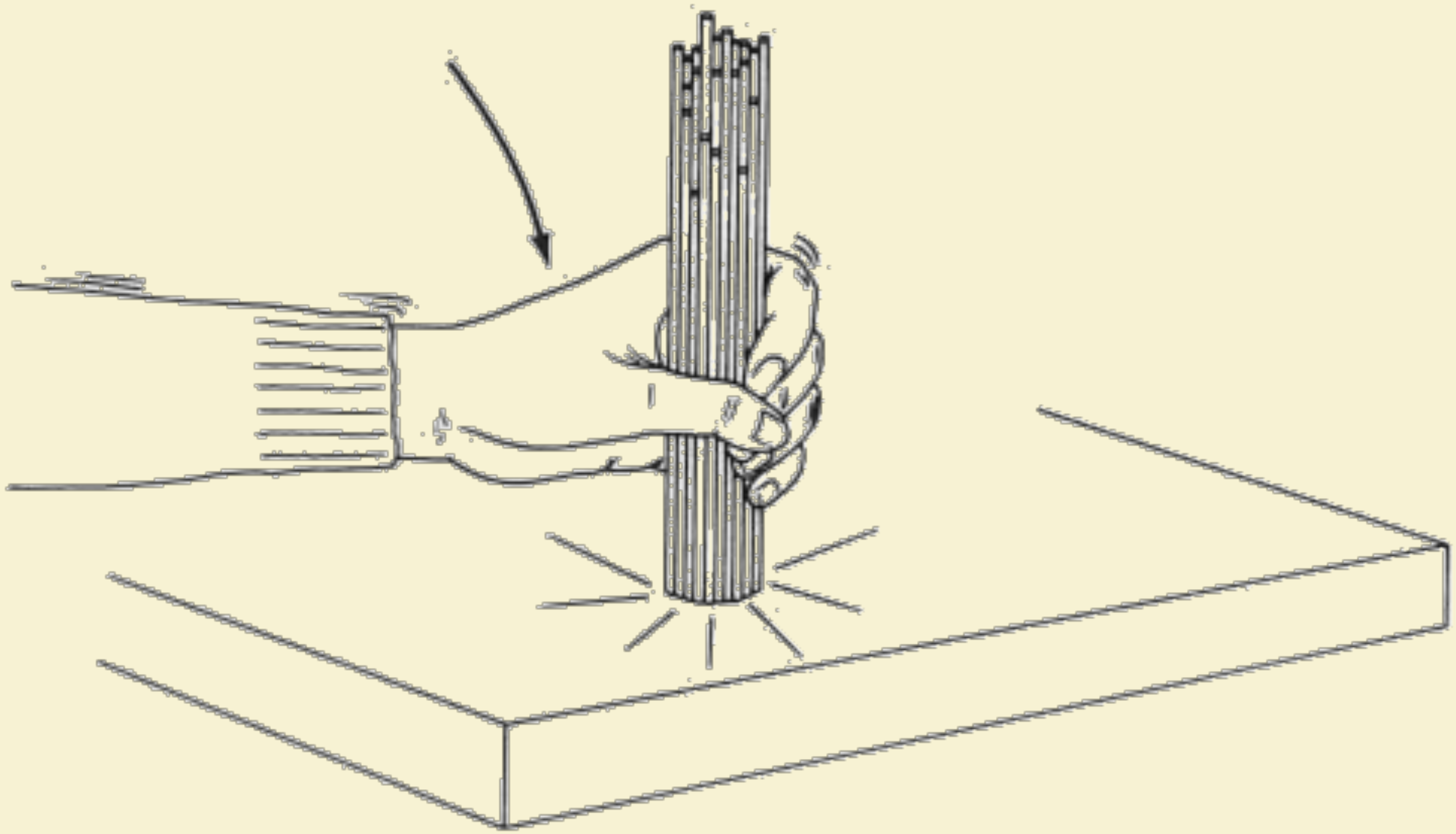
Soap Bubble Computer



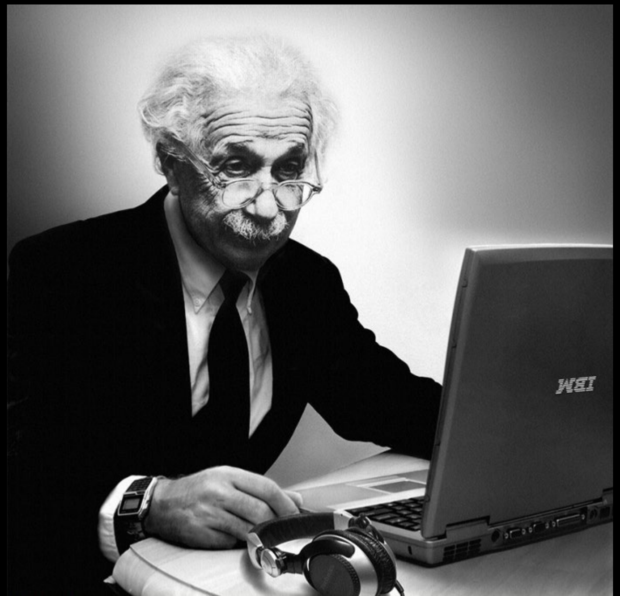
Protein Folding



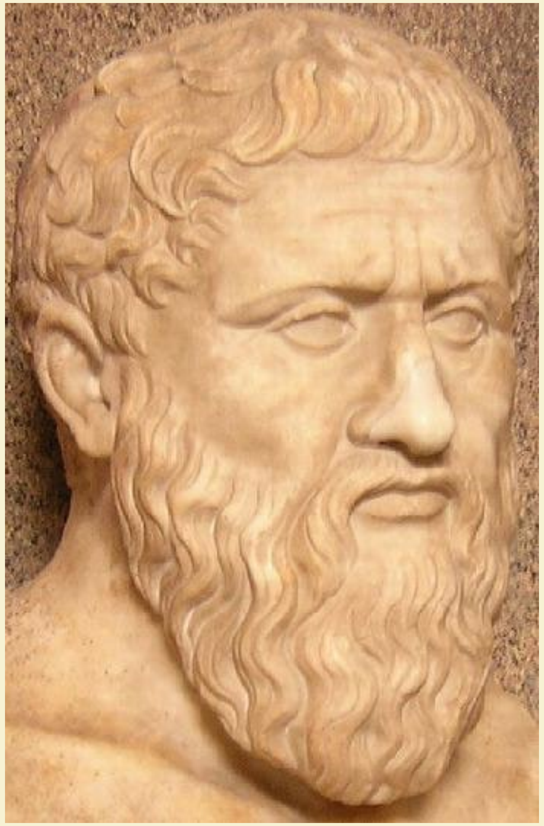
Spaghetti Sort

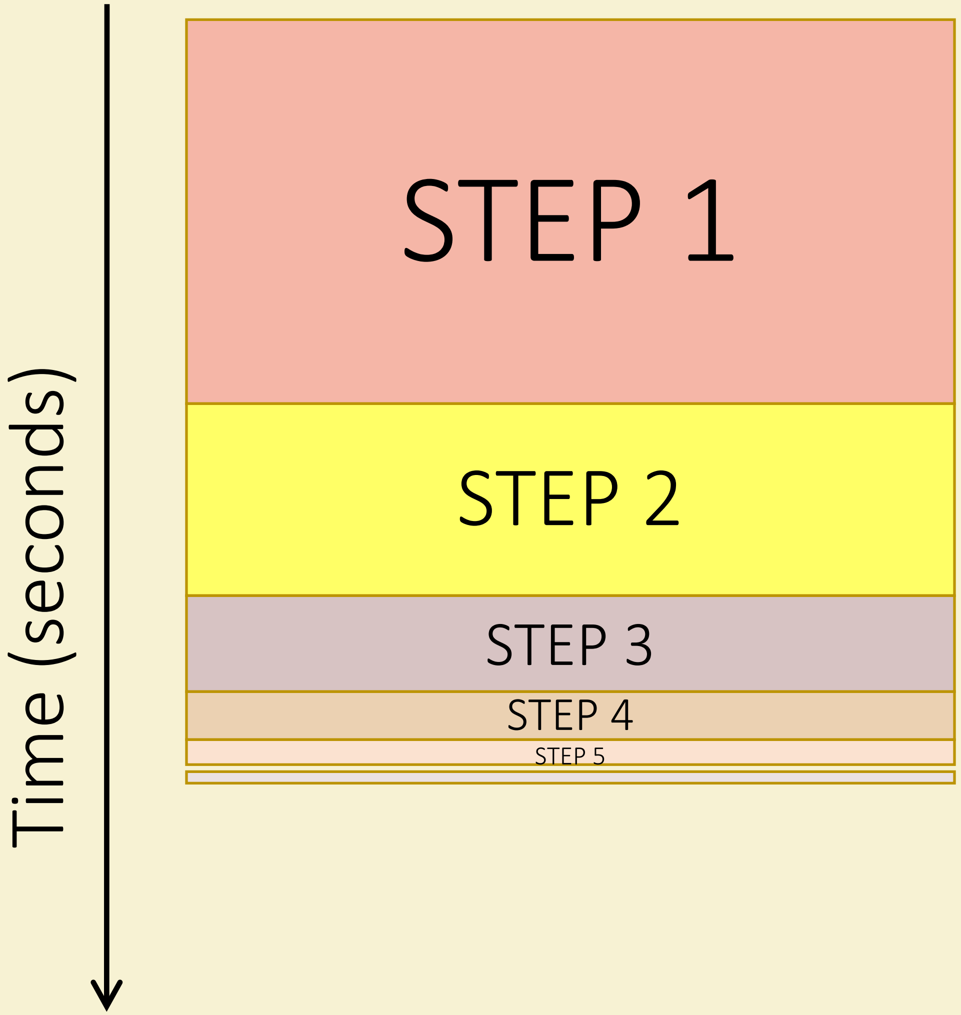


Relativity Computer
(cf. Malament and Hogarth)

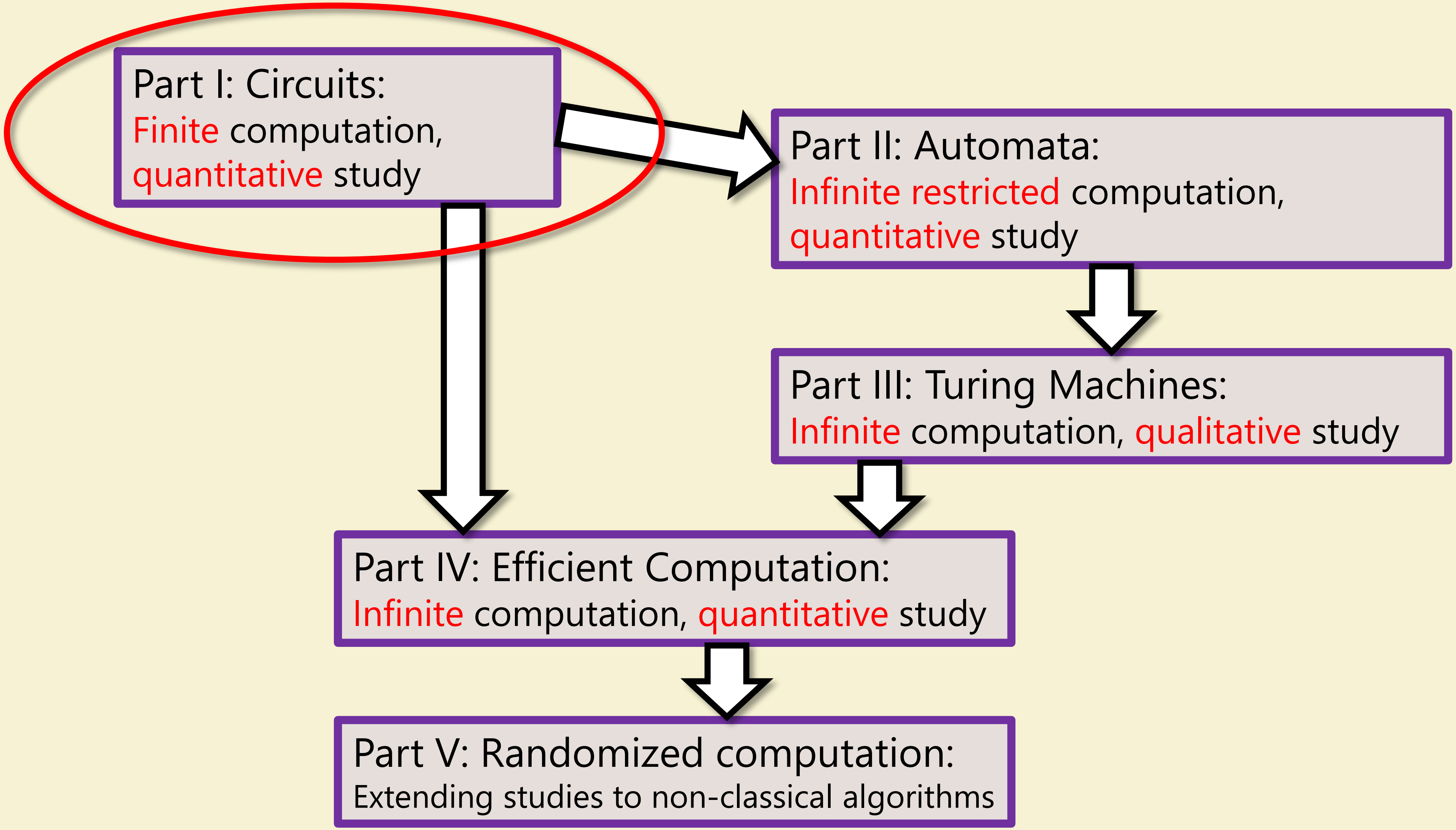


Zeno’s Computer





Where we are:


Part I: Circuits:
Finite computation, quantitative study
Part II: Automata:
Infinite restricted computation, quantitative study
Part III: Turing Machines:
Infinite computation, qualitative study
Part IV: Efficient Computation:
Infinite computation, quantitative study
Part V: Randomized computation:
Extending studies to non-classical algorithms

Where we are:

Part I: Circuits:
Finite computation, quantitative study

Part II: Automata:
Infinite restricted computation, quantitative study

Part III: Turing Machines:
Infinite computation, qualitative study

Part IV: Efficient Computation:
Infinite computation, quantitative study

Part V: Randomized computation:
Extending studies to non-classical algorithms

End of Lecture

If eligible,
Get Ready to Vote

- Today is National Voter Registration and Request your Ballot Day
- This year, young people are the largest voting bloc in the country.
  - Less than 50% of Harvard students voted in 2018.

- Visit **bit.ly/HVCpledge.** If eligible, make sure to check your voter registration (and make sure that the right addresses are listed) and either request a mail ballot or make a plan to vote in person.

- If you have already completed these steps, still fill out the form to double check! Sometimes people think they are registered when they're not! Not everyone is eligible to vote – encourage your friends to turn out and take action in other ways.

- **Questions**? Email voteschallenge@harvard.edu!

HARVARD VOTES CHALLENGE