Zander Kelley

Lecture 12 - Circuits

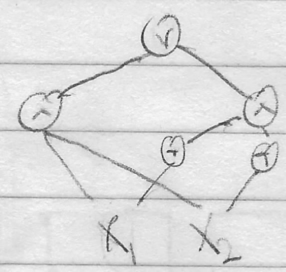
TMs = "Software"
Circuits = "hardware"

- Can always unroll software running in time T into hardware of size $T^{O(1)}$ ("Software" $\le$ "Hardware")
- Q: Is hardware of size poly(n) more powerful than software running in time poly(n) for problems of size n?
- ans: "tecnically yes but basically no (we think)."

def: a circuit C over $\{0,1\}^n$ is a DAG with vertices that are variables $(x_1, x_2, \dots, x_n)$ or gates $(\wedge, \vee, \neg)$.

- size(C) := # gates
- By specifying an output gate, C defines a function from $\{0,1\}^n$ to $\{0,1\}$
- size(f) := size of the smallest circuit implementing f.

$$Eq: \{0,1\}^2 \to \{0,1\}$$



$\wedge : \{0,1\}^2 \to \{0,1\}$
$\vee : \{0,1\}^2 \to \{0,1\}$
$\neg : \{0,1\} \to \{0,1\}$.

☆ For any $f: \{0,1\}^n \to \{0,1\}$, $\text{size}(f) \le O(n2^n)$ by encoding its truth table:

$$f(x) = \bigvee_{\substack{y \in \{0,1\}^n \\ f(y)=1}} \mathbb{1}(x=y) = \bigvee_{y: f(y)=1} \Big(Eq(x_1,y_1) \wedge Eq(x_2,y_2) \wedge \dots \wedge Eq(x_n,y_n)$$

• Q: What if we used a different gate set?

Let $G = \{g : \{0,1\}^k \to \{0,1\}\}$ for some $k = O(1)$, $k \geq 2$.

★ $size_G(C) = \Theta\left(size_{\{\wedge,\vee,\neg\}}(C)\right)$

pf: $size_{\{\wedge,\vee,\neg\}}(g) \leq O(k2^k) = O(1)$.

Let $f_L^{(n)} : \{0,1\}^n \to \{0,1\}$ be the indicator function

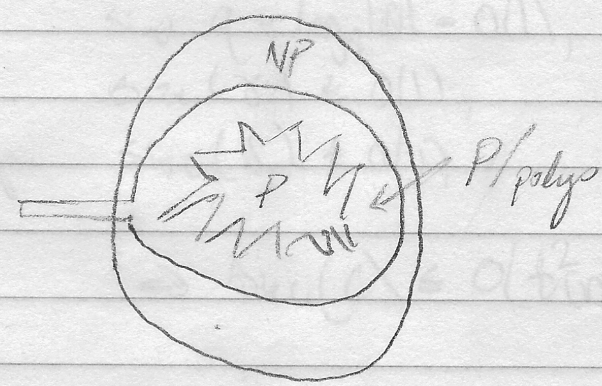
$$f_L^{(n)}(x) := \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{if } x \notin L \end{cases}$$

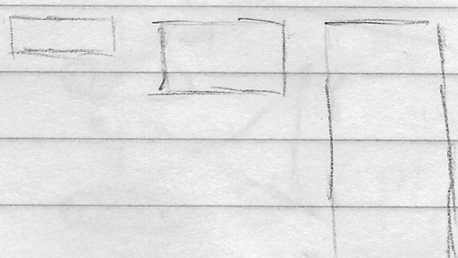
def: $SIZE(s(n)) = \{ L : size(f_L^{(n)}) \leq O(s(n)) \}$.

$P/poly := SIZE(poly(n))$

• Q: P vs P/poly?

• Q: NP vs P/poly?





★ $\mathrm{TIME}(t(n)) \subseteq \mathrm{SIZE}(s(n)^2)$ (in fact $\subseteq \mathrm{SIZE}(s \log s)$)

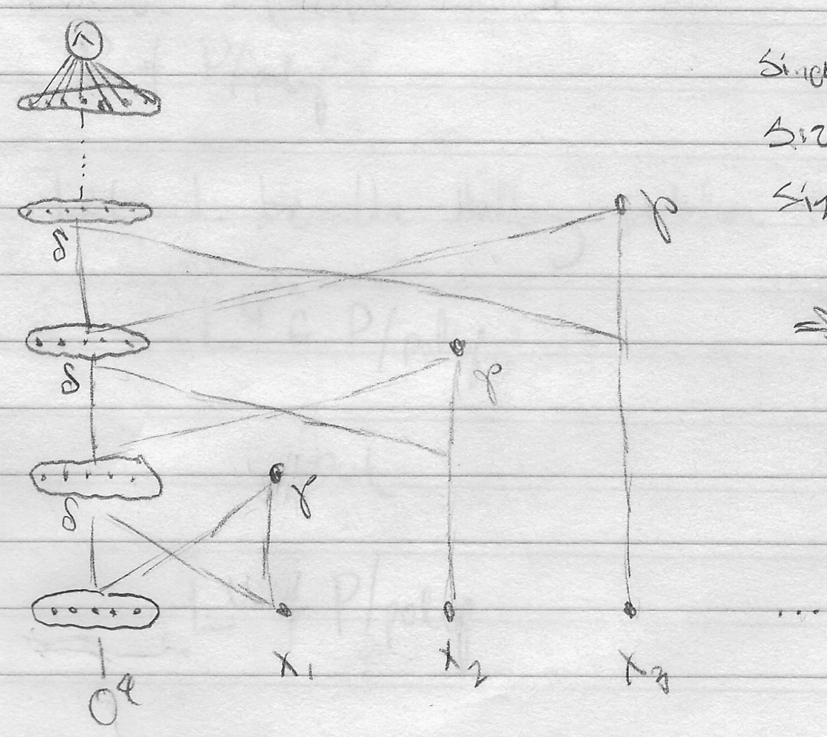
Cor: $P \subseteq P/poly$.

pf: Let $L \in \mathrm{TIME}(t(n))$; assume $\Sigma_L = \{0,1\}$ for simplicity. Let M be an oblivious TM for L running in time $O(t(n)^2)$.

M defined by $\delta : Q \times \{0,1\} \to Q$
$\gamma : Q \times \{0,1\} \to \{0,1\}$.

Encode Q as $\{0,1\}^q$.

Encode q_{accept} as 1^q, q_{start} as 0^q



Since $q = \log_2 |Q| = O(1)$,
$\mathrm{size}(\gamma) \le O(1)$,
$\mathrm{size}(\delta) \le O(1)$

$\Rightarrow \mathrm{size}(C) \le O(t^2(n))$

def: For $L \in \{0,1\}^*$, we define the unary version of L

$$L^u := \{ 1^n : (n \text{ in binary}) \in L \}.$$

☆ For any L, $L^u \in P/poly$.

pf: We use the advice function

$$a(n) = \begin{cases} 1 & \text{if } 1^n \in L^u \\ 0 & \text{if } 1^n \notin L^u \end{cases}$$

Our TM $M(x,a)$ just checks if x is of the form 1^n and outputs $a(n)$.

Cor: $P \neq P/poly$

pf: Let L be the Halting problem. Then

$$L^u \in P/poly$$

but

$$L^u \notin P/poly$$

def: Let $a : \mathbb{N} \to \{0,1\}^*$ be an "advice function".
A TM with advice (M, a) computes the function

$$x \mapsto M(x, a(|x|)).$$

$P/poly$ (advice) $:= \left\{ L : L \text{ can be decided by a TM w/ advice running in } poly(n) \text{ time.} \right\}$

☆ $P/poly$ (advice) $=$ $P/poly$ (circuit)

pf: $\supseteq$: give the circuit as advice and eval on x.
$\subseteq$: for a fixed input length n,
Implement $M(x,a)$ as a circuit $C(x,a)$.
Hardwire $a(n)$ into C.

Perspective on $P/poly$ vs NP:

$P/poly$ = "trusted advice, one per input size"
NP = "untrusted advice, one per input" (have to certify)

☆ Circuit-SAT is NP-hard

pf: Let F be the indicator function of some Language $L \in NP$.
Then

$$F(x) = \bigvee_{y \in \{0,1\}^{poly(n)}} f(x,y) \quad \text{for some poly-time function } f$$

$$= \bigvee_{y \in \{0,1\}^{poly(n)}} C(x,y) \quad \text{for some poly-size Circuit } C.$$

So $F(x) = 1$ iff $C_x(y) := C(x,y)$ is satisfyable

☆ 3-SAT is NP-hard.

Circuit $f(x)$

x_1

x_2

$\vdots$

x_n

$g_1 = x_1 \wedge x_3$

$g_2 = g_1 \vee x_4$

$\vdots$

$g_i = g_j \vee g_k \longrightarrow C_i := \text{"} y_i = y_j \vee y_k \text{"}.$

$\vdots$

$g_s = \ldots$

3-CNF $F(x,y)$

$$F(x,y) = \left(\bigwedge_i C_i\right) \wedge y_s$$

$f(x) = 1$ iff $F(x,y)$ is satisfyable in y
(In fact $F(x,y)$ is uniquely sat. in y)

Any function on 3 vars can be implemented as a 3-CNF (write $\neg f$ as a 3-DNF)

☆ 99% of functions require exponential-size circuits (and therefore also polynomial time to solve)

pf: Consider functions $f:\{0,1\}^n \to \{0,1\}$ (there are 2^{2^n}),

Let $\mathcal{F}(s) = \{ f : \text{size}(f) \leq s \}$.

Then $|\mathcal{F}(s)| \leq (s + 2s^2)^s \leq 2^{cs\log s}$.

So the fraction of functions w/ size $\leq s$ is

$$\frac{|\mathcal{F}(s)|}{2^{2^n}} \leq \frac{2^{cs\log s}}{2^{2^n}} = 2^{cs\log s - 2^n}.$$

Solving $2^{cs\log s - 2^n} = \frac{1}{100}$ gives

$$s\log s \geq \Omega(2^n)$$

$$\Rightarrow s \geq \Omega(2^n / n).$$

<u>Open Problem</u>: Prove that some explicit function requires circuits of size $s \geq 6n$.