Week 10

Announcements: HW3 due yesterday

T2 - graded + posted

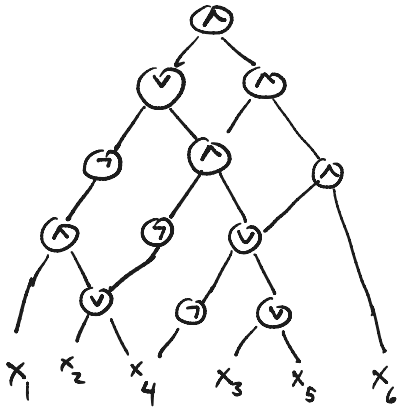
Next Fri NOV 20 : Test 3

Practice problems posted tonight/tmrw

Today: Intro to Circuit Complexity

Boolean Circuits

Boolean circuit over $\vee, \wedge, \neg$:



Defn Let $f : \{0,1\}^n \to \{0,1\}$.

A Boolean circuit C_n over inputs $x_1 \ldots x_n$ computes f if:

$$\forall \alpha \in \{0,1\}^n \quad C(\alpha) = f(\alpha)$$

The circuit complexity of f is min size circuit computing f

(size = # gates in circuit)

Defn A function $f : \{0,1\}^* \to \{0,1\}$ has circuit complexity $s(n)$ if $\forall n \geq 0$ f on $\{0,1\}^n$ is computed by a size $s(n)$ circuit

So a function $f : \{0,1\}^* \to \{0,1\}$ ⟵ or equivalently a language $L \subseteq \{0,1\}^*$

it has circuit complexity poly(n) if

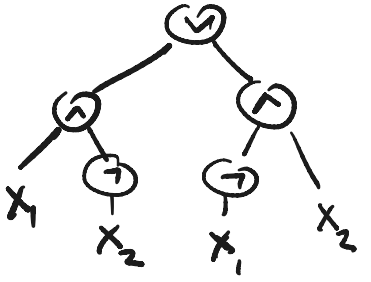
$\exists$ circuits $C_0, C_1, C_2, \ldots$

st $\forall n \geq 0$ C_n computes f on $\{0,1\}^n$ and $|C_n| = \text{poly}(n)$.

Ex $L = \{ x \in \{0,1\}^* \mid x \text{ has an odd number of 1's} \}$

C_1 : x_1

C_2 :



C_n : $(C_{n-1} \wedge \neg x_n) \vee (\neg C_{n-1} \wedge x_n)$

Claim 1 $L \in P \Rightarrow L$ has poly sized circuits

Note: converse not true, since it may be difficult to construct the nth circuit C_n

New Defn: P/poly [Poly sized circuits with advice]

A Language $L \subseteq \{0,1\}^*$ is in P/poly if:

$\exists$ poly time TM A with 2 inputs: $(x, \alpha_{|x|})$

input

advice string

same string $\forall x, |x| = n$

A is polytime in $|x|$

Can view advice α_n as description of circuit C_n

New Defn : P/poly [Poly sized circuits with advice]

A Language $L \subseteq \{0,1\}^*$ is in P/poly if :

$\exists$ poly time TM A with 2 inputs : $(x, \alpha_{|x|})$

input

advice string
same string
$\forall x, |x| = n$

A is polytime in $|x|$

Can view advice α_n as description of circuit C_n

On input $\alpha \in \{0,1\}^n$, compute $C_n(\alpha)$ and output answer.

Lemma

① $L \in P \Rightarrow L \in P/poly$

~~② $L \in P/poly \Rightarrow L \in P$~~

Also we have the following theorem (Karp Lipton) that says although P/poly is nonuniform, so this makes it much stronger than P

[we say a language in P/poly that isn't even decidable]

this thm gives evidence that it is unlikely that $NP \subseteq P/poly$

Thm If $NP \subseteq P/poly$ then $PH = \Sigma_2^P$ (poly hierarchy collapses to 2nd level)

Well studied circuit classes

(Note all are dags, arbitrary fanout)

$$\text{poly-size } AC^0 \subseteq \text{poly-size } AC^0[2] \subseteq \text{poly-size } NC^1 \subseteq \text{poly-size } NC \subseteq P/poly$$

- AC^0: $O(1)$ depth unbded fanin OR_1 AND plus $\neg$
- $AC^0[2]$: same as AC^0 plus unbded fanin $\oplus_2$
- NC^1: fanin 2 $\vee, \wedge$ plus $\neg$ depth $O(\log n)$
- NC: fanin 2 $\wedge, \vee$ plus $\neg$ depth $\log^{O(1)} n$
- $P/poly$: fanin 2 $\wedge, \vee$ plus $\neg$ No depth restriction

Well studied circuit classes

(Note all are dags, arbitrary fanout)

$$AC^0 \subseteq AC^0[2] \subseteq NC_1 \subseteq NC \subseteq P/poly$$

AC^0 : Unbded fanin $\vee$, $\wedge$ gates, and $\neg$

Depth $O(1)$ – dont count $\neg$ towards depth
can wlog push all $\neg$ to leaves

Equivalent : depth 2 $\wedge, \vee$ and depth is max # of $\vee, \wedge$ alternations on paths from root to leaf

Size : number of wires

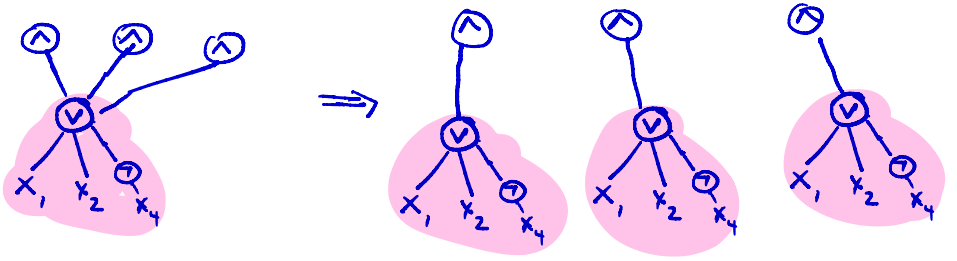
Note : we can assume for polysize AC^0 that circuit is a tree (fanout of all $\wedge, \vee$ gates is 1)

Example: CNF : Depth 2 $\wedge \circ \vee$

DNF : Depth 2 $\vee \circ \wedge$

$f \in$ size $s(n)$ AC^0_d $\Rightarrow$ $f \in$ size $\text{poly}(s(n))$ AC^0_d with fanout 1.

Pf: duplicate each $\vee$, $\wedge$ gate to get fanout 1, from leaves to root



New size: depth 1 $\rightarrow$ s^2

depth 2 $\rightarrow$ $s^2 \cdot s$

depth d $\rightarrow$ s^{d+1}

Example ADD(x,y): $x = x_1 \ldots x_n$ $y = y_1 \ldots y_n$ } binary rep'N

output $x+y$ in binary

Small AC^0 circuits for ADD?

① "gradeschool" method

$$\begin{array}{r} {\scriptstyle 1\ \ 1\ \ 1\ \ \ \ } \\ 01110 \quad x \\ 10111 \quad y \\ \hline 100101 \end{array}$$

sequential due to "carry"
so depth is n, size $O(n)$

② Depth 2 but exponential size:

DNF D_i for every output posn z_i

OR of all possible assignments to $x_i \ldots x_1, y_i \ldots y_1$
such that i^{th} posn of sum is 1

Example · ADD(x,y): $x = x_1 \ldots x_n$ $y = y_1 \ldots y_n$ } binary rep'N

③ "Carry Save Add'n": parallel-table, still size poly(n)

Idea: Carry bit $c_i = 1$ iff $\exists$ posn $j < i$ that generates a carry AND all positions between j and i propagate carry

generate a carry gen(i): $x_i \wedge y_i$

propagate a carry prop(i): $x_i \vee y_i$

$$c_i : \bigvee_{j=1}^{i-1} \left(\underbrace{x_j \wedge y_j}_{\text{j generates carry}} \wedge \underbrace{\bigwedge_{k=j+1}^{i-1} (x_k \vee y_k)}_{\text{all } j+1 \le k \le i-1 \text{ propagate carry}} \right)$$

i^{th} output bit $z_i \equiv x_i + y_i + c_i \bmod 2$: $(x_i \wedge y_i \wedge c_i) \vee (\bar{x}_i \wedge \bar{y}_i \wedge c_i) \vee (\bar{x}_i \wedge y_i \wedge \bar{c}_i) \vee (x_i \wedge \bar{y}_i \wedge \bar{c}_i)$

total depth is 4, size: $n \cdot O(n^2)$. Can you do better?

Well studied circuit classes (Note all are dags, arbitrary fanout)

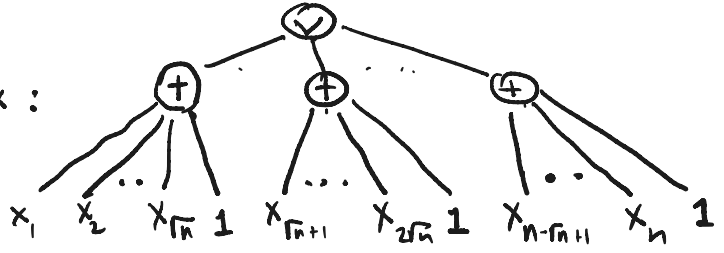
$$AC^0 \subseteq AC^0[2] \subseteq NC_1 \subseteq NC \subseteq P/poly$$

Just like AC^0 but now gates are unbded fanin $\vee$, $\wedge$, $\oplus$, plus $\neg$

Note: can assume NO $\neg$, NO $\wedge$ since:

$$\neg x \equiv x \oplus 1 \qquad A \wedge B \equiv \neg(\neg A \vee \neg B)$$

Ex:





← 1 iff some block has an odd number of 1's

$$AC^0 \subseteq AC^0[2] \subseteq NC_1 \subseteq NC \subseteq P/poly$$

NC_1 : fanin 2 $\vee, \wedge, \neg$. Depth $O(\log n)$

<u>Lemma</u> parity $(x_1 .. x_n) \in$ polysize NC_1.
also unbounded $\bigvee_m$ converts to binary $\vee$, depth $\log m$, same for $\bigwedge_m$
$\therefore AC^0[2] \subseteq NC_1$

<u>Other fxNs in polysize NC_1</u> :

1. majority $(x_1 .. x_n) = 1$ iff $\sum x_i \geq \lceil \frac{n}{2} \rceil$
2. Binary addition, multiplication, addition of n binary numbers, division
3. Sorting n numbers (each n bits long)

NC_1 is a rich class! Also Barrington thm showed
any $f \in NC_1$ can also be computed using <u>constant space</u>!

An NC_1 circuit for $Parity(x_1 \ldots x_n)$

Recursively construct a ckt for $parity(x_1 \ldots x_{\frac{n}{2}})$
and $parity(x_{\frac{n}{2}+1} \ldots x_n)$

$$Par_{1,n} = \left(\neg Par_{1,\frac{n}{2}} \wedge Par_{\frac{n}{2}+1,n}\right) \vee \left(Par_{1,\frac{n}{2}} \wedge \neg Par_{\frac{n}{2}+1,n}\right)$$

Written as formula: size n^2 depth $2\log n$

written as ckt : depth $2\log n$, size : n

An equivalent characterization of NC_1

$f \in$ polysize $NC_1 \iff f$ can be computed by polysized $\wedge, \vee, \neg$ formula

formula: special type of circuit where all gates have fanout 1

1. $f \in$ polysize formula (over $\wedge, \vee, \neg$) $\Rightarrow f \in$ polysized, depth $O(\log n)$ formula
   $\therefore f \in$ polysize NC_1

Idea: the formula for f is a polysized tree
It could be long and skinny! How to put in low depth?

$\frac{1}{3}$-$\frac{2}{3}$ trick: For any binary tree T of size S,
$\exists$ a subtree T' of T of size S', $\frac{S}{3} \leq S' \leq \frac{2S}{3}$

apply $\frac{1}{3}$-$\frac{2}{3}$ trick recursively to balance formula.

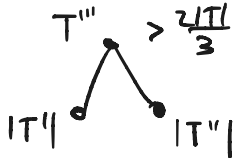
1. $f \in$ polysize formula (over $\wedge, \vee, \neg$) $\Rightarrow f \in$ polysized, depth $O(\log n)$ formula
$\therefore f \in$ polysize NC_1

Idea: the formula for f is a polysized tree
It could be long and skinny! How to put in low depth?

$\frac{1}{3}$-$\frac{2}{3}$ trick: For any binary tree T of size S,
$\exists$ a subtree T' of T of size S', $\frac{S}{3} \le S' \le \frac{2S}{3}$

* size = number of leaves

Proof Start at root of T. go Left if $|T_L| \ge |T_R|$, else go Right
Stop as soon as we reach subtree T', $|T| \le \frac{2|T|}{3}$

since 

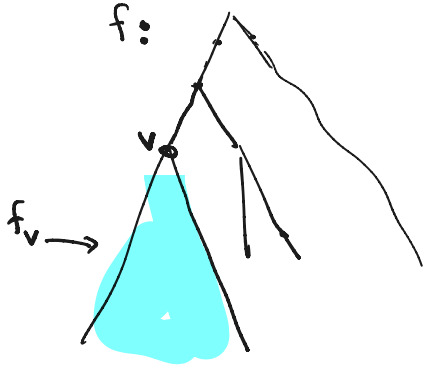
$|T'| + |T''| = |T'''|$

so $|T'| \ge \frac{|T'''|}{2} > \frac{2|T|}{3}\frac{1}{2} = \frac{|T|}{3}$

1. $f \in$ polysize formula (over $\wedge, \vee, \neg$) $\Rightarrow f \in$ polysized, depth $O(\log n)$ formula

$\therefore f \in$ polysize NC_1

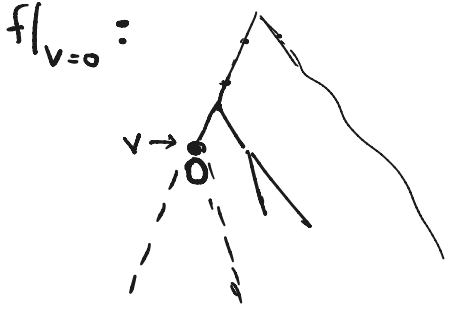
Recursively apply $\frac{1}{3}$-$\frac{2}{3}$ trick:



$$f(x) \equiv \left(f_v(x) \wedge f|_{v=1}(x)\right) \vee \left(\neg f_v(x) \wedge f|_{v=0}\right)$$

Says $f(x)=1$ if either $f_v(x)=1$ and $f|_{v=1}(x)=1$

OR $f_v(x)=0$ and $f|_{v=0}(x)=1$



Depth Recurrence

$$\underbrace{D(L)}_{\text{Depth of resulting tree when initial tree has } L \text{ leaves}} \leq D\left(\tfrac{2}{3}L\right) + 2 = O(\log_{3/2} L) = O(\log n)$$

when $L = n^{O(1)}$

An equivalent characterization of NC_1

$f \in$ polysize $NC_1 \iff f$ can be computed by polysized $\wedge, \vee, \neg$ formula

formula: special type of circuit where all gates have fanout 1

2. $f \in$ polysize $NC_1 \implies f \in$ polysize formula (over $\wedge, \vee, \neg$)

Pf idea: Top down convert depth $d = c\log n$, size $s = n^{O(1)}$ circuit into depth d, size $\text{poly}(s)$ circuit with fanout 2

gate is $u \wedge v$: recursively replace by $F_u \vee F_v$,
F_u, F_v recursively constructed, fanout 1

$u \vee v$: $F_u \vee F_v$

u : $\neg F_u$

An equivalent characterization of NC_1

2. $f \in$ polysize $NC_1 \implies f \in$ polysize formula (over $\wedge, \vee, \neg$)

Pf idea: Top down convert depth $d = c\log n$, size $s = n^{O(1)}$ circuit into depth d, size poly(s) circuit with fanout 2

* size: Top gate (level 0) $\longrightarrow$ one gate * uses fanin is ≤ 2

Level 1 $\longrightarrow$ 2 gates

Level 2 $\longrightarrow$ 4

Level d $\longrightarrow$ 2^d

$\therefore$ total size $1 + 2 + 4 + \cdots + 2^d \leq 2^{d+1}$

since $d = O(\log n)$ $\rightarrow$ size is $2^{O(\log n)} = \text{poly}(n)$

$$AC^0 \subseteq AC^0[2] \subseteq NC_1 \subseteq NC \subseteq P/poly$$

NC_i : binary $\wedge, \vee, \neg$, depth $O(\log^i n)$

$NC := \bigcup_i NC_i$

Problems in NC : Matching
Rank, determinant

$$AC^0 \subseteq AC^0[2] \subseteq NC_1 \subseteq NC \subseteq P/poly$$

P/poly : No restriction on depth.

Examples of problems in P/poly :

- All problems in NC
- All problems in BPP ! ← HW Question
  example : polynomial identity testing

Lower Bounds Currently here's what we can prove

① $\text{Parity}(x_1 \ldots x_n) = \sum_i x_i \bmod 2$

requires AC^0_d ckts of size $2^{n^{\Omega(\frac{1}{d})}}$ --

$(\forall d \ \exists \varepsilon < 1$ s.t. parity requires AC^0_d ckts of size $2^{\Omega(n^\varepsilon)})$

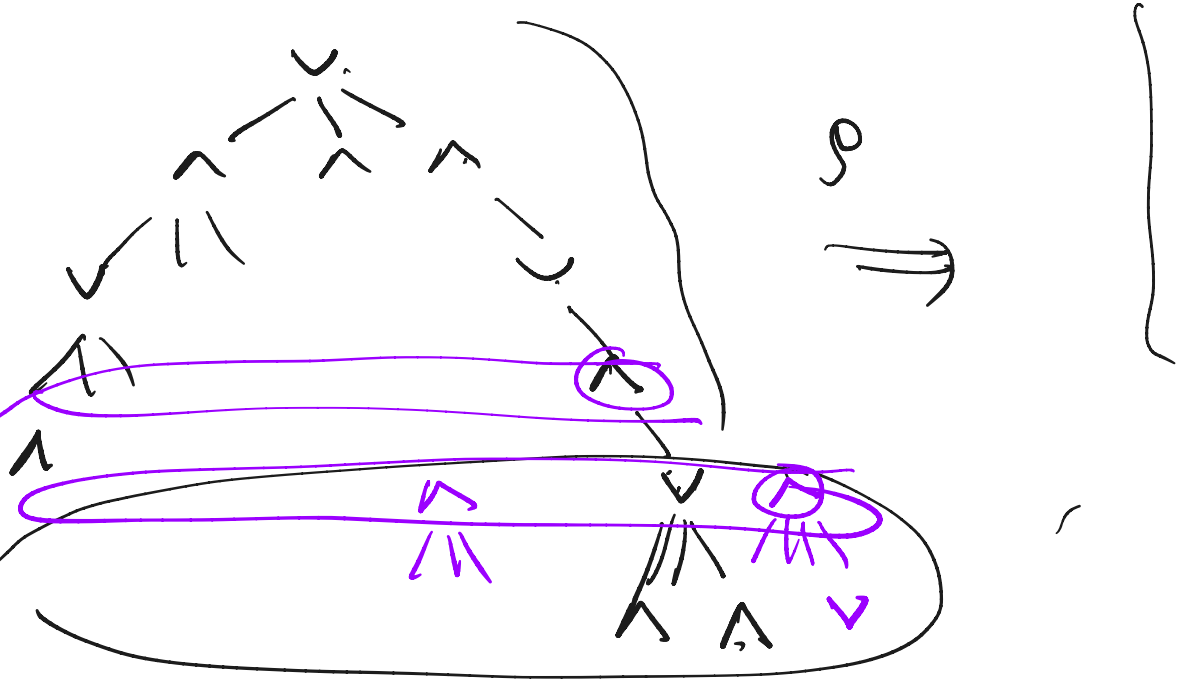
$\therefore$ polysize $AC^0 \not\subseteq$ polysize $AC^0[2]$

② $\text{Majority}(x_1 \ldots x_n)$ requires $AC^0[2]$ ckts of size $2^{n^{\Omega(\frac{1}{d})}}$

$\therefore AC^0[2] \not\subseteq NC_1$

* Note these are lower bounds for functions (parity, maj) that are in P.

switch lemma lets us:



$$AC^0 \subset AC^0[2] \subset NC_1 \subseteq NC \subseteq P/poly$$

Unknown : $NC \overset{?}{\subseteq} NC_1$

$P/poly \overset{?}{\subseteq} NC$

$NP \overset{?}{\subseteq} P/poly$

all believed to be false.

Easier q : $NP \subseteq NC_1$? also unknown

$EXP \subseteq NC_1$?

* Note : we can separate EXP from uniform NC_1 (and uniform $P/poly = P$)
using time hierarchy thm.
But for regular (nonuniform) defn of NC_1, open.

Monotone Circuit Lower Bounds

We can consider monotone Boolean functions $f : \{0,1\}^n \to \{0,1\}$.

f is monotone if $\forall x \leq y$ $(\leq : x_i = 1 \Rightarrow y_i = 1 \ \forall i)$ $f(x) = 1 \Rightarrow f(y) = 1$

Examples of monotone functions:

(1) Majority

(2) Clique_k: Input g represented as adjacency matrix
accept iff g has clique of size $\geq k$

(3) Bipartite (and nonbipartite) matching: Input g
accept iff g has a perfect matching

Monotone circuits: same but NO $\neg$ gate.

mP/poly : functions with monotone P/poly ckts of polysize, NO $\neg$'s

mNC^1, mNC defined same way.

Monotone Circuit Lower Bounds

Razborov famously showed the monotone version of NP $\not\subseteq$ P/poly:

Theorem $\text{Clique}_{\sqrt{n}/10}$ requires monotone circuits of size $\exp(n^{\varepsilon})$

Q: For monotone functions, can we get smaller circuits by allowing negations?

No: Matching $\in$ P/poly but any mP/poly circuit for matching requires exponential size*

*This is a new result. Razborov proved quasipoly LBs for mP/poly ckts in 80's. Exponential LB from last year: Cavalar, Göös, Riazonov, Sofronova, Sokolov '25

A Powerful Tool for Lower Bounds: Communication Complexity

Note: CC is important not just for circuit LBs;
it is used to prove impossibility results for many
types of models and problems (more later on applications)

2-player CC model: Input is 2 strings $x, y \in \{0,1\}^*$
Typically $|x| = |y| = n$. Want to compute $f(x,y)$

Alice gets x, Bob gets y [The function f is fixed]

They communicate thru a shared blackboard by
sending msgs back and forth.

Example: $\text{Parity}(x,y) = \sum_i x_i + \sum_i y_i \mod 2$

Alice

$x = 10011$

Bob

$y = 11011$



Alice computes & sends parity of x

Bob computes parity(y) & outputs Alice's bit + parity(y) mod 2

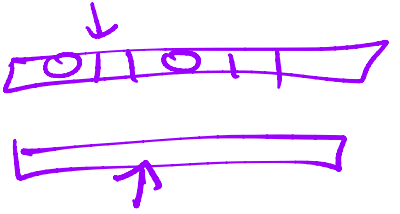


$CC(\text{Parity}_n) = O(1)$

Example 2 $EQ(x,y) = 1$ iff $x = y$ $\Omega(n)$

Alice x Bob y 0001 0001

Example 3 Output least index i where $(x_i \vee y_i) = 1$



Alice sends least index i s.t. $x_i = 1$ in binary or $n+1$ in binary OW

or 0 OW

$O(\log n)$

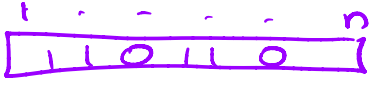
Example 4 output least index i s.t. $(x_i \wedge y_i) = 1$

or 0 if no such i exists

↗

variant of SET DISJOINTNESS DISJ

DISJ (x, y) : 1 iff $\exists i$ s.t. $x_i \wedge y_i = 1$

x 

y 

$S_x = \{1, 2, 4, 5\}$ $S_y = \{1, 2, 3\}$

↑ requires n cc $\Omega(n)$