LECT 04

Note Title

TODAY

1. NECIPORUK'S LOWER BOUND
2. BARRINGTON'S THEOREM
   (proof due to Ben-Or + Cleve)

Review

Last time: 4 Non-uniform Models of Computation

1. TMs with advice
2. Circuits
3. Branching ~~Programs~~
4. Formulae

Resources

- # bits of advice ;
- advice-TM. Time ;
- size
- depth
- width.

Counting bounds

if $\mathcal{F}$ is a family of functions

then $\exists\ f \in \mathcal{F}$ s.t.

$$\text{size}(f) \geq \Omega\left(\frac{\log(\mathcal{F})}{\log\log(\mathcal{F})}\right).$$

(Proof : # function of size $s \leq 2^{s \log s}$).

But : What is $NP \cap \{\{0,1\}^n \to \{0,1\}\}$?

Your answer here $\boxed{?}$

NECIPORUK'S THEOREM

$\exists f \in P$ s.t. BP-size $(f) \geq \left(\frac{n}{\log n}\right)^2$.

Proof Idea - 1 :

Will create function $f : \{0,1\}^k \times \{0,1\}^{n-k} \to \{0,1\}$

s.t. last $(n-k)$ bits really specify function on first k coordinates.

How: example

$f(i, \bar{x}) = x_i$ where

first i bits signify

index from $1 \ldots 2^k$

& $\bar{x}_1 \ldots x_{n-k} = 2^k$-bit

string.

$\forall\ \bar{x} \neq \bar{y} \qquad f_x(\cdot) \neq f_y(\cdot)$

where $f_x(i) = f(i, x)$.

Now BP for f gives BP for

f_x for every x.

$\#\ x's = 2^{n-k} \approx 2^n \Rightarrow \text{BP.size}(f) \geq |\{f_x\}|$

$$\geq \frac{n}{\log n}$$

But this is sub-linear ... how to improve?

Idea 2: for $S \subseteq [n]$

$\text{BP-size}_S(f)$ = # edges labelled with literals in S

Above proof actually implies

$$\text{BP-size}_{\{1..k\}}(f) \geq \frac{n}{\log n}$$

$\uparrow$

$\log n$ variables

Can we repeat this for other blocks.

Well ... not for same f, but

different one

__Function:__ DISTINCT? $(\overbrace{\underbrace{X_1 \ldots X_{1k}}_{U_1}}^{S_1}, \overbrace{\underbrace{X_{21} \ldots X_{2k}}_{U_2}}^{S_2}, \ldots, \overbrace{\underbrace{X_{\ell 1} \ldots X_{\ell k}}_{U_\ell}}^{S_\ell})$

$$\text{DISTINCT?}(U_1, \ldots U_\ell) = 1 \quad \text{if} \quad \forall i \neq j \quad U_i \neq U_j$$
$$= 0 \quad \text{o.w.}$$

__Claim:__ $\forall i$, # functions

$$\left| \left\{ f_{a_1 \ldots a_\ell} (U_i) = \text{Distinct?}(a_1, \ldots, U_i, \ldots a_\ell) \right\} \right| \geq \binom{2^k}{\ell - 1}$$

$$\geq \left(\frac{2^k}{\ell} \right)^\ell$$

Claim: $\text{BP-size}_{S_i}(\text{Distinct}) \geq \frac{n}{\log n}$

Claim: $\text{BP-SIZE}(f) \geq \sum_i \text{BP-size}_{S_i}(f)$

Putting Claims together: get

$$\text{BP-Size}(\text{Distinctness}) \geq \ell^2 k - \ell^2 \log \ell$$

Setting $k = 2\log \ell$ & $n = \ell k$

get

$$\geq \Omega\left(\frac{n^2}{\log^2 n}\right)$$

$\boxtimes$ (NECIPORUK)

BARRINGTON (BEN-OR + CLEVE):

Motivation:

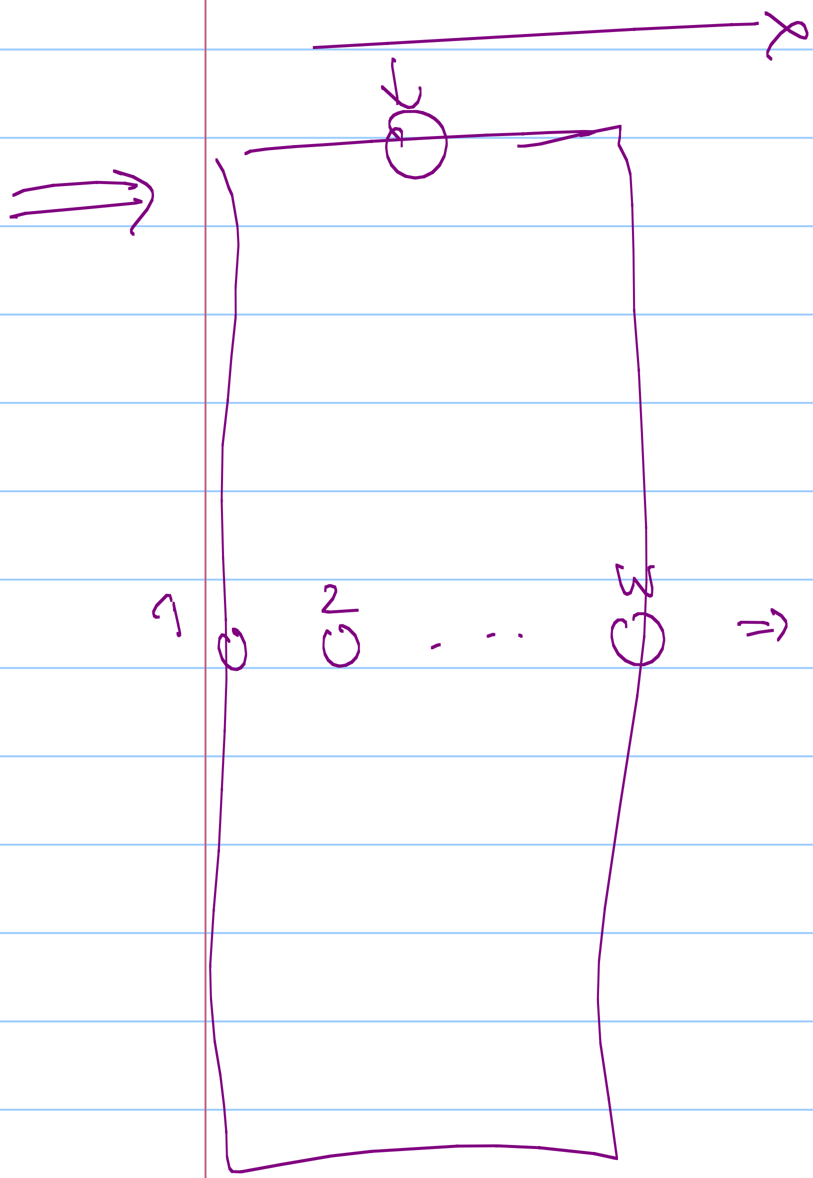
- Can we use non-uniformity to prove $P \neq L$?
- Maybe we can argue that "simple" functions don't have small width BPs.
- But every CNF / DNF formula has width - 3 BP ... of exponential size ...
- Really need to show that no poly-size BP exists for some function in P.
- Natural candidate: $\text{MAJORITY}(x_1 \ldots x_n)$

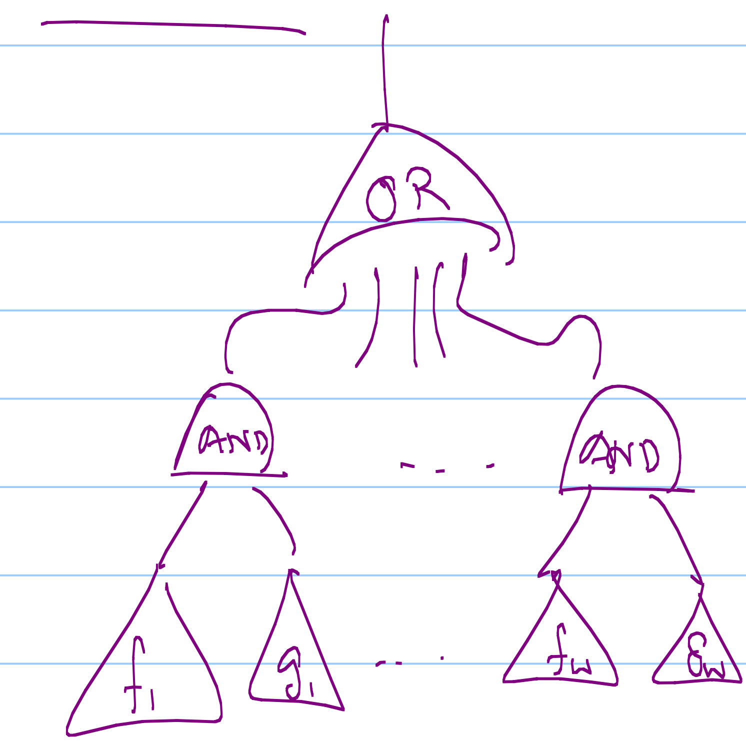
$$= 1 \iff \sum x_i \geq \frac{n}{2}.$$

BARRINGTON'S THEOREM:

f has $O(1)$ width poly size BP

$\Longleftrightarrow$ f has log-depth formula (over any finite basis)





(other bases similar)

In above

$f_i = 1$ if top b.p. reaches i^{th} state in middle level

$g_i = 1$ if bottom b.p. accepts starting at middle level.

$\Longleftarrow$ Much harder (unexpected)

Ben-Or + Cleve's proof : Simple idea.

"Strong Induction"

Register Machines

Given by ℓ registers $R_1 \ldots R_\ell$

& S instructions

I_1

I_2

$\vdots$

I_s

I_j : of the form $R_i \leftarrow R_j + R_k * R_\ell$

or $R_i \leftarrow R_j \pm X_k * R_\ell$

Register Machine Computes $f(x_1 \ldots x_n)$

if it maps:

$$(R_0 \ldots R_\ell) \to (R_1 \ldots R_{\ell-1},\ R_\ell + f(\,) \cdot R_\ell)$$

Strong Hypothesis: if f has depth d $\{$AND, NOT$\}$ formula then

$\exists$ size 4^d, 3 register machine

Prop: f has size S, ℓ-Register m/c

$\Rightarrow$ f has size $O(S)$, 2^ℓ width BP

(8 in our case)

Proof:

f computed by $M_1 = \begin{matrix} I_1 \\ \vdots \\ I_s \end{matrix}$

$\Rightarrow$ $(1-f)$ computed by

$$\begin{matrix} I_1' \\ \vdots \\ I_s' \end{matrix} \qquad I_j' = \overline{I}_j \text{ replace } + \text{ by } -.$$

$I_{s+1} = R_\ell \leftarrow R_\ell + R_1$

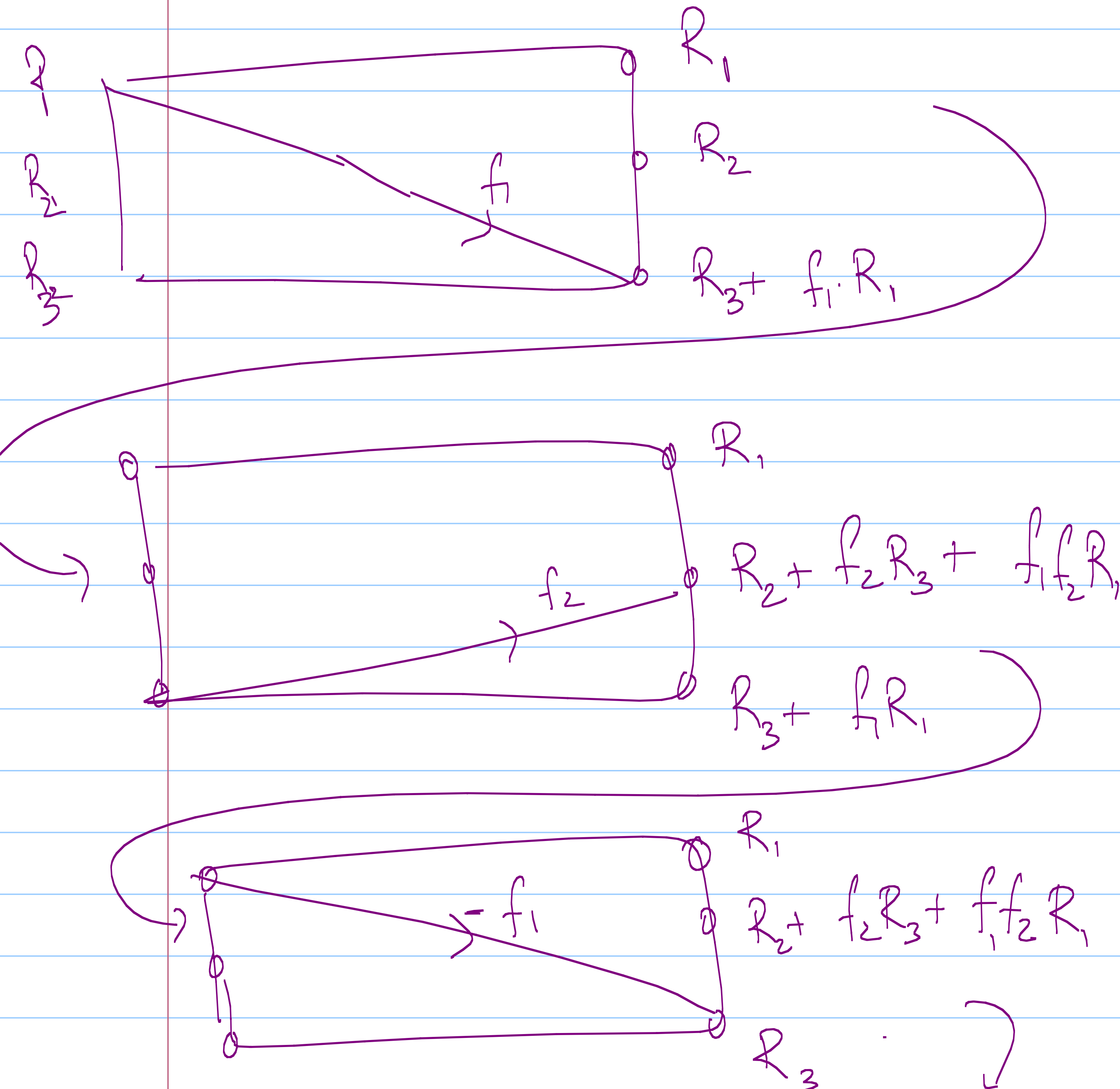
(previously R_ℓ had $R_\ell^0 - f R_1^0$

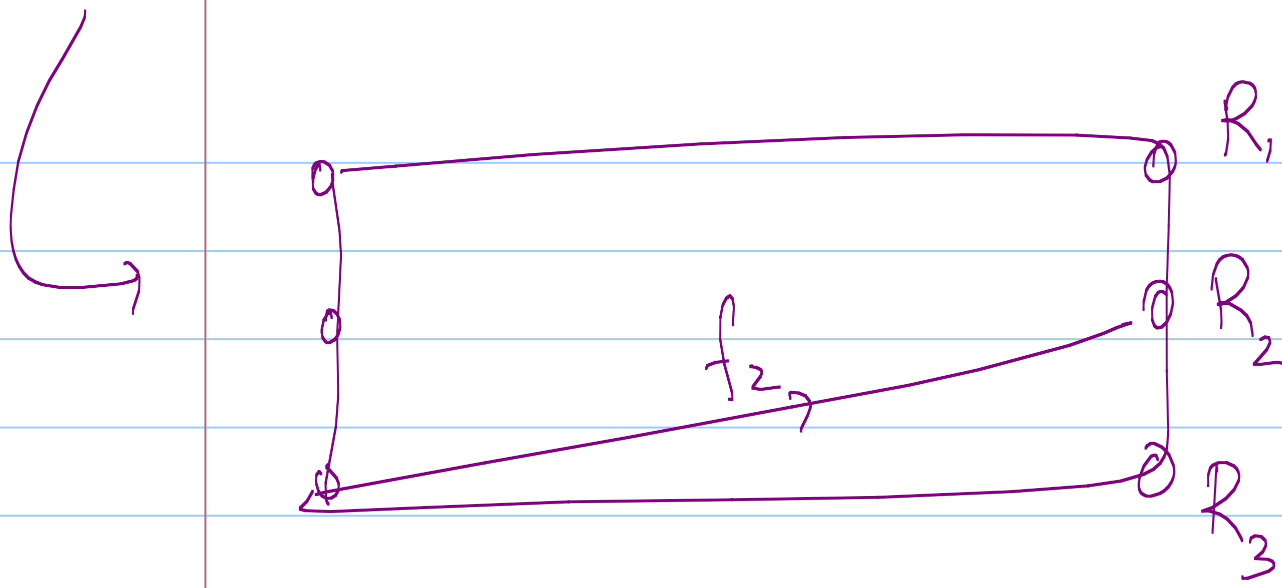
& R_1 " R_1^0

Now: $R_\ell \leftarrow R_\ell^0 + (1-f) R_1^0$)

Interesting Case

$$f = f_1 \wedge f_2$$





$$\underline{\underline{R_2 + f_1 f_2 R_1}}$$

$\uparrow$

~~8~~