Today

1. Neciporuk's Lower Bound

2. Barrington's Theorem
   (proof due to Ben-Or + Cleve)

Review

Last time: A Non-uniform Models of Computation

1. Trees with advice
2. Circuits
3. Branching Programs
4. Formulae
Resources
- # bits of advice
- advice Thm. Time
- size
- depth
- width

Counting bounds

If $I$ is a family of functions
then $\exists f \in I$ s.t.

$$\text{size}(f) \geq \Delta \left( \frac{\log (|I|)}{\log \log (|I|)} \right).$$
Proof: First, we will prove...
on first $k$ coordinates.

How: Example

$$f(i, \overline{x}) = X_i$$ where

first $i$ bits signify index from $1 \ldots 2^k$

& $\overline{x}_1 \ldots x_{n-2} = 2^k \cdot \text{bit string}$

$\forall \overline{x} \neq \overline{y}$

$$f_x(\cdot) \neq f_y(\cdot)$$

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

Now BP for $f$ gives BP for $f_x$ for every $x$.

$$\# x's = 2^{n-k} < 2^n \implies \text{BP:size}(f) \geq \frac{1}{2} 5 \times 3$$
But this is sub-linear ... how to improve?

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

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

Above proof actually implies

$\text{BP-size}_{\{1, \ldots, k\}}(f) \geq \frac{n}{\log n}$
Can we repeat this for other blocks.

Well ... not for same \( f \), but different one

**Function:** \( \text{DISTINCT?} \left( X_1, \ldots, X_{k_1}, X_{21}, \ldots, X_{2e}, \ldots, X_{l1}, X_{l2}, \ldots, X_{le} \right) \)

\[ \text{DISTINCT?} \left( U_1, \ldots, U_e \right) = 1 \text{ if } \forall i \neq j \quad U_i \neq U_j \]

\[ = 0 \text{ otherwise.} \]

**Claim:** \( \forall i, \# \text{ functions} \)

\[ \left| \{ f \mid \left( U_i \right) = \text{DISTINCT}\left( a_1, \ldots, U_i, \ldots, a_e \right) \} \right| \geq \left( \frac{2^k \cdot l}{e} \right) ^{l-1} \]
Claim: $\text{BP-size}_{\text{Dist-ind}}(f) \geq \frac{n}{\log n}$

Claim: $\text{BP-size}(f) \geq \sum_{i} \text{BP-size}_{s_i}(f)$

Putting Claims Together: get

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

Letting $k = 2 \log \ell$ and $N = \ell^k$

get

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

$\Box$ (NECIPORUK)
**Barrington (Ben-Or + Cleve):**

Motivation:
- Can we use non-uniformity to prove $P = L$?

- 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: $\text{Majority}(x_1, \ldots, x_n) = 1 \iff \exists x: x \geq \frac{n}{2}$.
Barrington's Theorem:

If $f$ has $O(n)$ width poly size BP

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

$\Rightarrow$

(Other bases similar)
In above

\[ f_i = 1 \text{ if top b.p. reaches } i^{th} \text{ state in mobile level} \]

\[ g_i = 1 \text{ if bottom b.p. accepts starting at mobile level.} \]

\[ \leftarrow \text{ Much harder (unexpected)} \]

Ben Or + Cleve's proof: Simple idea.

"Strong Induction"
Register Machines

Given by \( l \) registers \( R_1, \ldots, R_l \)

\( l \) \( S \) instructions

\( I_1 \)
\( I_2 \)
\[ \vdots \]
\( I_s \)

\( I_j \): of the form \( R_i \leftarrow R_j + R_k \times R_l \)
or \( R_i \leftarrow R_j + X R_l \)
Register Machine computes \( f(x_1, \ldots, x_n) \)

\[ f \Rightarrow (R_0, \ldots, R_{e-1}) \rightarrow (R_0, \ldots, R_{e-1}, R_e + f(\cdot) \cdot R_e) \]

**Strong Hypothesis:** if \( f \) has depth 1

\[ \{ \text{AND, NOT} \} \] formula true

\[ \exists \text{ size } 4^d, 3 \text{ register machine} \]

**Prop:** if \( f \) has size \( S \), \( l \)-register m/c

\[ \Rightarrow f \text{ has size } O(S), 2^l \text{ width BP} \]

(8 in our case)
Proof:

\[
\begin{align*}
&\text{\( f \) computed by } M_i = \overline{I}_i \\
\Rightarrow &\text{ \((1-f)\) computed by } \overline{I}_s \\
\therefore &\overline{I}_s = \overline{I}_s \text{ replace by } -
\end{align*}
\]

\[
\overline{I}_{S+1} = R_e \leftarrow R_e + R_c
\]

(Previously \( R_e \) had \( R_e^0 - fR_c \) & \( R_c \leftarrow R_c^0 \)

Now: \( R_e \leftarrow R_e + (1-f)R_c^0 \)
Interesting one

\[ f = f_1 \land f_2 \]

\[
\begin{align*}
R_1 & = f_1 \cdot R_1 \\
R_2 & = f_2 \cdot R_2 \\
R_3 & = f_3 \cdot R_3
\end{align*}
\]

\[
\begin{align*}
R_1 & = f_1 \cdot R_1 \\
R_2 & = f_2 \cdot R_2 + f_1 \cdot f_2 \cdot R_1 \\
R_3 & = f_3 \cdot R_3
\end{align*}
\]

\[
\begin{align*}
R_1 & = f_1 \cdot R_1 \\
R_2 & = f_2 \cdot R_2 + f_1 \cdot f_2 \cdot R_1 \\
R_3 & = f_3 \cdot R_3
\end{align*}
\]
\[ R_1 = R_2 + f_{27} f_{28} R_3 \]