Today

- Non-Uniform Computation
  \( \text{P/} Poly, \text{ Circuits}, \text{ Branching Programs, Formulae} \)
  \[ P/Poly = \text{Poly-size Circuits}. \]

- Some easy counting based bounds

- Nepiorka's lower bound on formula size.

---

Notation for \( L \leq 50, 1^* \)

Let \( L_n = L \cap 50, 1^n \)

Non-Uniform = Not uniform w.r.t. input length

i.e. different alg. \( A_n \) for \( L_n \) (for \( n \))
Various Models

Advice Turing Machines: Consider two right

Turing machine $M$ with advice string

Sequence $\vec{a} = (a_1, a_2, a_3, \ldots, a_n, \ldots)$

language accepted by $M$ with advice

sequence $\vec{a}$ is

$L(M, \vec{a}) = \{ x | M(x, a_i, x_i) \text{ accepts}\}$

Resources: In addition to the usual ones, count

$L(n) = \lceil an \rceil$.

if $M, \vec{a}$ runs in time $t(n)$ on first right of

length $n$ then $L(M, \vec{a}) \in \text{DTIME}(t(n))$.
\[
P/\text{poly} = \bigcup_{\text{poly}} \text{DTIME}(p(n)) / p(n)
\]

Important class.

Similarly, one can talk about NP/\text{poly}, L/\text{poly}, ... .

\underline{Circuits (over basis of gates } g_1, \ldots, g_k) \\
\text{(example binary AND, OR, + unary NOT)}

\text{Circuit: DAG with}

- \text{sources: input gates : in-degree = 0 :}
- \text{labeled } x_1, \ldots, x_n
- \underline{\text{distinct}}
remaining vertices: computing gates: labelled with basis function

not distinct

in-degree = in-deg of function

out-degree = arbitrary

Some m of vertices also designated output gates

computes function mapping \((x_1, \ldots, x_n) \rightarrow (y_1, \ldots, y_m)\)

by letting input vertices be labelled with input values; computing gates be labelled with value of function at input nodes
Circuit $C$ with $n$ inputs
1 output
decide $L_n$ if $\forall x \ L(x) = C(x)$.

Example

\[ \text{Principal} \]

Measures:

Size $= \# \text{ edges in DAG}$

Circuit complexity of $L$ is size of $C_n$ as a function of $n$. 
Depth = longest path in DAG.

Easy to show "EXERCISES"

1. Circuit complexity changes only by constant factors as basis changes from one finite (computable) set to another.

2. $L \in \text{DIME}(t(n))/\ell(n))$

   $\Rightarrow L$ has circuit complexity $O((t(n) + \ell(n))^2)$

   (Turing Machine tableau, YADA YADA YADA)

3. $L$ has circuit complexity $s(n)$

   $\Rightarrow L \in \text{DIME}(s(n)^2)/s(n)$
   (ditto)
Conclusion: $P/poly = \text{poly-size circuits}$.

Circuit complexity of function $f = \text{Nonuniform time}$

What about Non-uniform Space?

Need another model

**Branching Programs** (only defining for decision problems)

BP also given by DAG

- One source = start node
- Two sinks labelled 0 & 1;
- Remaining nodes have out-degree 2 labelled by complementary literals.
BP computes function naturally e.g.

Attempts if $x_1 + x_2 + x_3 = 1 \pmod{3}$

BP layered if vertices partitioned into layers $L_1, \ldots, L_{k-1}$.

Edges of $L_i$ go to $L_i$ or $L_{i+1}$.

$\text{Width} (BP) = \max \left\{ |L_i| \right\}$. 
Principal measure

\[ \log(\text{Width}) \leq \text{non-uniform space} \]

Size (BP) \leq \# \text{ nodes} \leq \text{Width}

Finally

Formula \equiv \text{Circuit in which all nodes except input have out-degree 1;}

(Why consider: formalization of colloquial term formula)
Some Exercises

- Formula Size $= 2^{O(\text{Formula depth})}$ (for finite basis) [trivial]

- $F\text{-DEPTH}(f)$ in basis $B$, $= \Theta(F\text{-DEPTH}(f) \text{ in basis } B_2)$

- For $f : \{0,1\}^n \to \{0,1\}$

$\quad F\text{-depth}(f) = O(\log(F\text{-SIZE}(f)))$ ! [little harder]

- $F\text{-SIZE}(f)$ in basis $B$, $\leq (F\text{-SIZE}(f) \text{ in basis } B_2)^{O(1)}$
Formulae vs. Branching Programs

- $\text{BP-size}(f) \leq O\left(\text{F-size}(f) \text{ in basis (AND, OR, NOT})\right)$

(for other bases, I'm not sure)

Branching Program Size vs. Circuit Size

- $\text{Ckt-size}(f) \leq O\left(\text{BP-size}\right)$ [Simple]

To Prove \(\text{NP} \neq \text{P}\)

suffices to give function in \(\text{NP} - \text{P}^{\text{poly}}\)

(seems more combinatorial)
To prove $P = L$

...suffices to give function in $P$ that does not have poly-size B.P.

Unfortunately all the above open.

Best known $C^k_1$-lower bound for function in $NP$ (in basis AND, OR, NOT) is $\sim 4.5n$.

Best known $BP$-size lower bound is $o(n^2)$.
Counting + Elementary Arguments

1. Every function $f$ has $F$-size in $(\text{AND, OR, NOT basis})$ $O(2^n)$

Proof: $f(x_1, \ldots, x_n)$

$= (x_1 = 0) \land f_0(x_2, \ldots, x_n)$

$\lor (x_1 = 1) \land f_1(x_2, \ldots, x_n)$

$\in O(2^n)$

2. \# functions with Ckt.-size $S \leq S$ 

(f-size / BP-size)

3. \# functions with Ckt.-size $\geq \frac{2^n}{n}$
\[
4. \quad \text{Ckt.-Size} (S) \leq \text{Ckt.-Size} (S \cdot \log S)
\]

etc.

Neciporuk’s BP lower bound

Then: \[ f (\text{clt+ distinctness}) \text{ needing } n^2 \text{ BP size,} \]

Idea: Counting + Restriction \[ \log^2 n \]

Proof:

For \[ S \subseteq [n] \]

\[
\text{SIZE}_S (BP) = \# \text{ gates with out-edges labelled by clts of } S.
\]

If \[ S_1, \ldots, S_k = \text{ partition of } [n] \]

Then \[ \text{SIZE} (BP) \geq \sum_{i=1}^{k} \text{SIZE}_{S_i} (BP) \]
Will create function 
\[ f \left( \frac{x_1 - x_e}{s_1}, \frac{x_1}{s_1} - \frac{x_2}{s_2}, \ldots, \frac{x_{1^k} - x_e}{s_{1^k}} \right) \]

Note: \[ \text{BP-size}_{S_i}(f) = \Omega(k^2 \cdot k \log k) \]

Theorem follows.

How to lower bound \( \text{BP-size}_{S_i}(f) \) ?

Distinct

1. Count # functions on \( S_i \) obtained by fixing bits of \( \sum_{j \neq i} S_j \).

2. log of above is a l.b. on \( \text{BP-size}_{S_i}(f) \)

Why? ----
... Because every such function has

\[ \text{BP of size } \text{BP-size } S_i \] (f)

let \[ y'(x'_1, \ldots, x'_k) = f(x'_1, \ldots, x'_k, 0, 1, 1, 0, \ldots) \]

then take \( \text{BP (f)} \)

If \( y = 0 \) drop y edge & collapse A & C
Repeat: Only edges remaining are
How to create \( f \) s.t. \( \# S_i \)

is large?

\[
f(x_1, \ldots, x_k, w_1, w_2, \ldots, w_k) = 1 \text{ if } \exists i \neq j \text{ s.t. } w_i = w_j
\]

Now by fixing \( w_2, \ldots, w_k \) distinct in \([1, 2^k]\)

got \( \binom{2^k}{k} \) distinct functions

\[
\log \left( \binom{2^k}{k} \right) \geq k \log k - k \log k \]