Today

Non-uniform classes = TMs with advice

Circuits, Formulae, Branching programs.

Formulae \leq BPs \leq Branching programs

Depth and Width; Formula depth to size relationship.

Quadratic lower bound on formula size (Neciporuk).

Barrington’s result on bounded-width branching programs.

Non-uniform computation

One way to view non-uniform computation, is in the form of a two input Turing machine.

- $M(x, y)$ in class say $\mathcal{C}$.
- $M$ decides $L$ with advice $a = \langle a_1, \ldots, a_n \ldots, \rangle$ if for every $x \in \{0, 1\}^n$
  $M(x, a_n) = 1 \iff x \in L$.
- If $|a_n| = \ell(n)$, then $L$ said to be in $\mathcal{C}/\ell$.

Most important subcase: $P/poly = polysize circuits = non-uniform polytime$.

Recall other non-uniform measures

- Formula: Circuits whose DAG is a tree.
- Branching programs: Diff. model of computation, where nodes represent decisions to be taken, and end point determines answer.
- Circuits: Fully powerful non-uniformity.

Formulae, BPs, Circuits

- $f$ has formula size $s(n)$ implies $f$ has BP size at most $O(s(n))$.
  - By induction: $f_1$ and $f_2$, $f_1$ or $f_2$, $\overline{f_1}$.
- $f$ has BP size $s(n)$ implies $f$ has circuit size at most $O(s(n))$.
  - Induction on size of BP; note that declaring any other node of BP to be start node gives BP of smaller size. Assume there is a circuit of size $c.(s-1)$ computing all other functions. Now to compute start node, will add $c$ extra nodes. Note BP $f$ can be written as $x_1 \cdot f_1 + \overline{x_1} \cdot f_2$. 

Brief History

- Formula size lower bounds: Best known over any basis $\Omega(n^3)$ or so. Will show an $\Omega(n^2)$ lower bound today.
- BP size - slightly behind.
- Circuit size: Only $4.5n$, over AND, OR, NOT basis.
- Easy to show existence of functions that need size $2^{\Omega(n)}$.

Neciporuk’s lower bound for Formula size

- $f$ is the Distinctness function.
  - $f$ takes $n\ell$ bits as input. (Will later set $\ell = 2 \log_2 n$.)
  - View input as $X_1, \ldots, X_n$ where $X_i = \langle x_{i,1}, \ldots, x_{i,\ell} \rangle$.
  - $f(X_1, \ldots, X_n) = 1$ iff $\exists i \neq j$ s.t. $X_i = X_j$.
  - Thm: $f$ needs formulae of size at least $\Omega(n^2)$, provided $\ell \geq 2 \log_2 n$.

Proof

- Basic ideas: Counting, and restrictions.
- Claim 1: Number of leaves involving variables from $X_i$ is at least $\Omega(n)$.
  * Assume o.w. Let $\#$ leaves $= k$.
  * Then $\#$ formulae obtained by restricting other variables to 0/1 is at most $2^{O(k)}$.
  * But $\#$ functions obtained by other restrictions is at least $\binom{2^{\ell}}{n-1}$.

Moving on

- Size lower bounds very weak in unrestricted case. So restrict models and then prove strong lower bounds.
- Example: consider monotone functions (changing input bit from 0 to 1 cannot change output from 1 to 0), and prove lower bounds for circuits without negation.
- Example: Restrict "depth" of circuit/BP, "width" of branching program.
Depth, Width

- Depth of DAG = length of longest path in DAG.
- Width of Layered DAG:
  * DAG is layered if vertices are partitioned into layers $L_1, \ldots, L_k$ and all edges run from $L_i$ to $L_{i+1}$.
  * Width of Layered DAG is max number of vertices in Layer.
- Can define width of unlayered DAG also, but not as clean.

Basic observations about depth

- Depth of circuit = parallel time; Can even allow unbounded fan-in OR/AND, to simulate CRCW models.
- Circuit depth = formula depth.
- Formula depth = $\Theta(\log \text{Formula size})$.
- Thus Formula size $\text{poly}(n) = \log n$ depth.
- In upcoming lecture: Show limitations of poly size circuit of constant depth (with unbounded fan-in OR/AND).
- Depth of BP = time. (Size = space).

Basic observations about width

- Width of BP = non-uniform space.
- Fair amount of intuition obtained by unravelling DFAs. (DFA unravelling leads to Layered read-once branching program.) Thus $O(1)$-width BPs can count modulo $O(1)$.
- Early belief: possibly can’t do anything else. Can’t compute majority?
- Easy to rule out poly size width-2 BPs computing majority. Hard result: width-3 BPs can’t compute majority. Hope in the 80s ... will eventually rule out all $O(1)$ width (or even sublinear width).

- Major recent breakthrough: Ajtai shows explicit functions requiring nearly linear space to be done in nearly linear time. (Won’t cover.)
Barrington’s Theorem

- Every function with poly size formula can be computed with width 5 bp.

- Ben-Or Cleve Proof:
  - wlog: prove for log-depth arithmetic formula. (Why does this suffice? Exercise!)
  - Prove for 3-register machines = width 8 bps. (slightly weaker).

Define Register Machines

Ben-Or + Cleve proof of Barrington