Lecture 14: NC\(^1\) and Barrington’s Theorem

![Branching Program Diagram]
Theorem 14.1 The set of problems accepted by uniform (polynomial size) branching programs is $\text{DSPACE}\log n$.

\[
\text{BranchingPrograms} = \text{L}
\]

Proof:
BranchingPrograms $\subseteq \text{L}$: just keep track of where you are!

$L \subseteq \text{BranchingPrograms}$:
Let $M$ be a $\text{DSPACE}\log n$ Turing machine.

The computation graph of $M$ on some variable input $x_1 \cdots x_n$ is a branching program! \hfill $\square$
Proposition 14.2  The set of problems accepted by uniform, bounded-width branching programs is contained in $\text{NC}^1$.

Proof: hw 10
Bounded Width Branching Programs look very much like finite automata.

\[ \text{MAJ} = \{ w \in \{0, 1\}^* \mid w \text{ contains more than } |w|/2 \text{ “1”s} \} \]

**Natural Conjecture:**

\[ \text{MAJ} \not\subseteq \text{Bounded Width BPs} \]
$S_5$ is the permutation group on 5 objects.

$$\alpha = (12345), \quad \beta = (13542) \in S_5$$

$$[\alpha, \beta] = \alpha \beta \alpha^{-1} \beta^{-1}$$

$$= (12345)(13542)(54321)(24531)$$

$$= (13254)$$
Definition 14.3 A width 5 Branching Program, $B$, 5-cycle recognizes $S$ iff for some 5-cycle $\sigma$,

- For $x \in S$, $B(x) = \sigma$
- For $x \notin S$, $B(x) = e$

Lemma 14.4 Let $S_i = \{x \in \{0, 1\}^n \mid x_i = 1\}$. $S_i$ can be 5-cycle recognized.

Lemma 14.5 If $S$ is 5-cycle recognized, then so is $\overline{S}$
Lemma 14.6 If $S$ is 5-cycle recognized using 5-cycle $\sigma$, then $S$ can be 5-cycle recognized using 5-cycle $\tau$.

Proof: Every two 5-cycles are conjugates, i.e.,

$$\exists \theta \in S_5)(\tau = \theta^{-1} \sigma \theta)$$

□

Lemma 14.7 If $S$ and $T$ can be 5-cycle recognized by branching programs $B$ and $C$, then $S \cap T$ can be 5-cycle recognized by a branching program of size $2(|B| + |C|)$.

Proof:

$$B \ C \ B^{-1} \ C^{-1}$$

□
Theorem 14.8 (Barrington’s Theorem)

\[ Bounded \text{ Width Branching Programs} = \text{NC}^1 \]

**Proof:**

Given an NC\(^1\) circuit, simulate it using the above lemmas.

We multiply the size of the branching programs by 4 as we go up one level.

Total size is \(4^{O(\log n)} = n^{O(1)}\)