Lecture 19: Circuit Complexity

Real computers are built from gates.

Circuit complexity is a low-level model of computation.

Circuits are directed acyclic graphs. Inputs are placed at the leaves. Signals proceed up toward the root, $r$.

Straight-line code: gates are not reused.

Let $S \subseteq \{0, 1\}^*$ be a decision problem.

Let, $C_1, C_2, C_3, \ldots$ be a circuit family.

$C_n$ has $n$ input bits and one output bit $r$.

**Def:** \( \{C_i\}_{i \in \mathbb{N}} \text{ computes } S \) iff for all $n$ and for all $w \in \{0, 1\}^n$,

\[
  w \in S \iff C_{|w|}(w) = 1.
\]
"not" gates are pushed down to bottom

\[
\text{Depth} = \text{parallel time}
\]

\[
\text{Number of gates} = \text{computational work} = \text{sequential time}
\]

\[
\text{Width} = \text{max number of gates at any level} = \text{amount of hardware in corresponding parallel machine}
\]
Circuit Complexity Classes

$S \subseteq \{0, 1\}^*$ is in NC[$t(n)$], AC[t(n)], ThC[t(n)], iff exists uniform circuit family, $C_1, C_2, \ldots$, s.t.

1. For all $w \in \{0, 1\}^*$, $w \in S \iff C_{|w|}(w) = 1$
2. $\text{depth}(C_n) = O(t(n))$; $|C_n| \leq n^{O(1)}$
3. The gates of $C_n$ consist of,

**NC**

- bounded fan–in
- and, or gates

**AC**

- unbounded fan–in
- and, or gates, or gates

**ThC**

- unbounded fan–in
- threshold gates

![Diagram of circuit complexity classes](image-url)
**Notation:** for $i = 0, 1, \ldots$, \[ NC^i = NC[(\log n)^i]; \]

\[ AC^i = AC(\log n)^i; \quad \text{ThC}^i = \text{ThC}(\log n)^i \]

We will see that the following inclusions hold:

\[
\begin{align*}
AC^0 & \subseteq \text{ThC}^0 \subseteq NC^1 \subseteq L \subseteq NL \subseteq AC^1 \\
AC^1 & \subseteq \text{ThC}^1 \subseteq NC^2 \subseteq AC^2 \\
AC^2 & \subseteq \text{ThC}^2 \subseteq NC^3 \subseteq AC^3 \\
\vdots & \subseteq \vdots \subseteq \vdots & \subseteq \vdots \\
\end{align*}
\]

Thus:

\[ NC = \bigcup_{i=0}^{\infty} NC^i = \bigcup_{i=0}^{\infty} AC^i = \bigcup_{i=0}^{\infty} \text{ThC}^i \]
Uniform means that the map, $f : 1^n \mapsto C_n$ is very easy. $f \in F(L)$; $f \in F(FO)$

Each $C_i$ is an instance of the same program.
**Prop:** Every regular language is in NC¹.

**Proof:** DFA $D = (\Sigma, Q, \delta, s, F)$. Build circuits: $C_1, C_2, \ldots$,

\[
\begin{align*}
&f_1(q) = \delta(q, w_i); & w \in \mathcal{L}(D) \iff f_{1n}(s) \in F
\end{align*}
\]
Thm: \( \text{FO} = \text{AC}^0 \)

Example: \[ \phi \equiv \exists x \forall y \exists z (M(x, y, z)) \]
Prop: For $i = 0, 1, \ldots$, 
\[ \text{NC}^i \subseteq \text{AC}^i \subseteq \text{ThC}^i \subseteq \text{NC}^{i+1} \]

Proof: All inclusions except $\text{ThC}^i \subseteq \text{NC}^{i+1}$ are clear.

$$ \text{MAJ} = \{ w \in \{0, 1\}^* \mid \text{w has more than } |w|/2 \text{ “1”s} \} \in \text{ThC}^0 $$

Lemma: $\text{MAJ} \in \text{NC}^1$

(and the same for any other threshold gate).
Try to build an NC$^1$ circuit for majority by adding the $n$ input bits via a full binary tree of height $\log n$.

**Problem:** the sums being added have more and more bits; still want to add them in constant depth.
Solution: Ambiguous Notation

Binary representation; but with digits: 0, 1, 2, 3

$$\begin{align*}
3213 &= 3 \cdot 2^3 + 2 \cdot 2^2 + 1 \cdot 2^1 + 3 \cdot 2^0 = 37 \\
3221 &= 3 \cdot 2^3 + 2 \cdot 2^2 + 2 \cdot 2^1 + 1 \cdot 2^0 = 37
\end{align*}$$

Lemma: Ambiguous Notation Addition is in NC^0

Example:

```
carries:  3  2  2  3
       3  2  1  3
    + 3  2  1  3
       3  2  2  1  0
```

The carry from column \(i\) is determined by columns \(i\) and \(i + 1\): use the largest carry we are sure to get.
Translating from ambiguous to binary, is just addition, thus first-order, thus $AC^0$, and thus $NC^1$. 
Arithmetic Hierarchy

Recursive

 FO(∃(N)) r.e.
 FO(∀(N)) r.e.
 FO(¬) co-r.e.
 FO-SAT co-r.e.
 FO-VALID r.e.

 FO(N) co-r.e.
 Halt r.e.

Primitive Recursive

 SuccinctHornSAT EXPTIME complete

 EXPTIME

 SO(LFP) SO[2^{n^{O(1)}}]

 QSAT PSPACE complete

 PSPACE

 FO[2^{n^{O(1)}}] FO(PFP) SO(TC) SO[n^{O(1)}]

 PTIME Hierarchy

 co-NP complete SAT

 SAT

 co-NP SO∀

 NP SO∃

 NP ∩ co-NP

 FO[\sqrt{n^{O(1)}}] P complete

 FO(LFP) SO(Horn)

 Horn-SAT P

 FO[\sqrt{n^{O(1)}}] “truly feasible” NC

 FO[\sqrt{n^{O(1)}}] “truly feasible” AC^1

 FO[\sqrt{n^{O(1)}}] “truly feasible” sAC^1

 FO(DTC) 2COLOR L comp.

 FO(TC) SO(Krom) 2SAT NL comp.

 FO(CFL) FO(TC) SO(Krom) 2SAT NL comp.

 FO(TC) SO(Krom) NL

 FO(DTC) 2COLOR L comp.

 FO(REGULAR) NC^1

 FO(COUNT) ThC^0

 FO AC^0

 LOGTIME Hierarchy

 AC^0