Computational Complexity — tutorial 6

Boolean circuits

Theory: understand Lecture 4 and the statement of theorem in Lecture 5.

We define the following complexity classes for $k \in \{0, 1, 2, \dots\}$:

- AC^k: problems which can be solved by circuits with polynomial size and depth at most $O(\log^k n)$ where n is the number of inputs.

  Also: $\mathsf{AC} = \mathsf{AC}^0 \cup \mathsf{AC}^1 \cup \mathsf{AC}^2 \cup \mathsf{AC}^3 \cup \dots$.
- NC^k: problems which can be solved by circuits with polynomial size and depth at most $O(\log^k n)$ in which all gates have **at most** 2 **inputs** (have fan-in $\leqslant 2$).

  Also: $\mathsf{NC} = \mathsf{NC}^0 \cup \mathsf{NC}^1 \cup \mathsf{NC}^2 \cup \mathsf{NC}^3 \cup \dots$.

Also, these classes have **uniform** variants (sometimes written $\mathsf{u\text{-}AC}^k$, $\mathsf{u\text{-}AC}$, $\mathsf{u\text{-}NC}^k$, $\mathsf{u\text{-}NC}$) where we only consider circuits which can be generated in logarithmic space. Formally, there must exist a logarithmic space algorithm which, given an input 1^n (n ones), outputs the description of the circuit with n inputs. Of course, $\mathsf{u\text{-}AC}^k \subseteq \mathsf{AC}^k$ and $\mathsf{u\text{-}NC}^k \subseteq \mathsf{NC}^k$. Warning: some people/sources define AC^k and NC^k as the uniform variants of the classes themselves.

We know that for each $k \geqslant 0$, we have $\mathsf{NC}^k \subseteq \mathsf{AC}^k$ (by definition) and $\mathsf{AC}^k \subseteq \mathsf{NC}^{k+1}$ (an OR gate with m inputs can be replaced with a binary tree of OR gates of depth $O(\log m)$ with m leaves, and m is a polynomial of n; similarly for AND gates). Hence, $\mathsf{AC} = \mathsf{NC}$. An equivalent description follows for uniform complexity classes.

[trivial] $\mathsf{NC}^0 \subsetneq \mathsf{AC}^0$ since circuits with bounded fan-in and bounded depth cannot even compute the OR of all inputs (assuming sufficiently many inputs), which can be done using a single gate with unbounded fan-in.

[Furst, Saxe, Sipser 1984] The Parity problem (given n inputs, decide if an odd number of them is true) is **not** in AC^0. It's obviously in NC^1 (and in $\mathsf{u\text{-}NC}^1$). Hence:

$$\mathsf{NC}^0 \subsetneq \mathsf{AC}^0 \subsetneq \mathsf{NC}^1 \subseteq \mathsf{AC}^1 \subseteq \mathsf{NC}^2 \subseteq \mathsf{AC}^2 \subseteq \mathsf{NC}^3 \subseteq \mathsf{AC}^3 \subseteq \dots \subseteq \mathsf{NC} = \mathsf{AC}.$$

Similarly:

$$\mathsf{u\text{-}NC}^0 \subsetneq \mathsf{u\text{-}AC}^0 \subsetneq \mathsf{u\text{-}NC}^1 \subseteq \mathsf{u\text{-}AC}^1 \subseteq \mathsf{u\text{-}NC}^2 \subseteq \mathsf{u\text{-}AC}^2 \subseteq \mathsf{u\text{-}NC}^3 \subseteq \mathsf{u\text{-}AC}^3 \subseteq \dots \subseteq \mathsf{u\text{-}NC} = \mathsf{u\text{-}AC} \subseteq \mathsf{P}.$$

Open problems:

- Is $\mathsf{u\text{-}NC} \stackrel{?}{=} \mathsf{P}$? (Equivalently, $\mathsf{u\text{-}AC} \stackrel{?}{=} \mathsf{P}$.)
- Can we prove $\mathsf{NC}^k \stackrel{?}{\subsetneq} \mathsf{AC}^k$ or $\mathsf{AC}^k \stackrel{?}{\subsetneq} \mathsf{NC}^{k+1}$ for some $k \geqslant 1$? (similar for uniform variants)
- Describe the hierarchy $\mathsf{NC}^1 \subseteq \mathsf{NC}^2 \subseteq \mathsf{NC}^3 \subseteq \dots \subseteq \mathsf{NC}$. We only know that if $\mathsf{NC}^i = \mathsf{NC}^{i+1}$ for some i, then $\mathsf{NC}^i = \mathsf{NC}^{i+1} = \mathsf{NC}^{i+2} = \dots = \mathsf{NC}$. Otherwise, all inclusions are strict. (similar for uniform variants)

1. We consider the ADDITION problem: given two n-bit numbers, produce their sum (formally, construct a function $\mathsf{add} : \{0,1\}^{2n} \to \{0,1\}^{n+1}$ which adds n low bits of the argument to n high bits of the argument).

(a) Find a circuit of size $O(n)$ solving ADDITION[1].

(b) Prove that ADDITION is in AC^0 (i.e., find a circuit of constant depth, polynomial size and potentially unbounded fan-in).

(c) Find a circuit of size $O(n)$, depth $O(\log n)$, and bounded fan-in solving ADDITION[2].

2. Prove the following fact: for every circuit with n gates and depth d with arbitrary AND, OR & NOT gates, there exists an equivalent circuit with $\leqslant 2n$ gates and depth $\leqslant d$ where NOT gates can only be applied directly to the inputs of the circuit (but AND & OR gates can still be used arbitrarily).

3. We consider the MULTIPLICATION problem: given two n-bit numbers, produce their product.

(a) Prove that MULTIPLICATION is in AC^1.

(b) *(a bit tricky)* Prove that MULTIPLICATION is in NC^1.

Hint: define a function add3to2 *mapping triples of integers* (x, y, z) *into pairs of integers* (x', y') *such that* $x + y + z = x' + y'$*. Can it be implemented in constant depth using only unary and binary gates?*

(c) Prove that MULTIPLICATION is not in AC^0.

Hint: solve PARITY *using* MULTIPLICATION *as a black-box.*

4. We consider the relations between the regular languages and circuit complexity classes.

(a) Prove that there exists a regular language not in AC^0.

(b) Prove that there exists a language in AC^0 which is not regular.

(c) Prove that all regular languages are in NC^1 (even in $\mathsf{u\text{-}NC}^1$).

5. *(a bit harder)* Construct a Boolean circuit with n inputs $x_1, x_2, \ldots, x_n$, n outputs $y_1, y_2, \ldots, y_n$, $O(n)$ gates **and wires** (gates can have unbounded fan-in), and constant depth which computes the prefix OR-sums of the sequence. Formally, for each $i \in \{1, 2, \ldots, n\}$, we want the i-th output y_i to be equal to $x_1 \vee x_2 \vee \cdots \vee x_i$.

The problem was a homework assignment in the 2015/2016 course, though a significant hint was added to the statement of the original problem. Click [here] if you're looking for one.

[1] This is also called a ripple-carry adder.
[2] This is also called a carry-lookahead adder.