Boolean Circuits

Andrei Bulatov

Boolean Gates

To model parallel computation we use extremely simple processors.

The three types of them can only compute the three logic connectives:

- **AND**
- **OR**
- **NOT**

inputs

outputs
Boolean Circuits

Definition
A Boolean circuit is a collection of gates and inputs connected by wires such that:

• every gate input is connected to exactly one circuit input or one gate output

• every gate output except for one, called the circuit output, is connected to at least one gate input

• cycles are not permitted

Example Boolean Circuit

Inputs (at top) $X, Y$
Computes (at bottom) $X \text{xor} Y$

We use the fact that $X \text{xor} Y = (X \lor Y) \land (\neg X \lor \neg Y)$
Boolean Circuit for Addition

Suppose we have two 2-bit numbers: $X_1X_0$ and $Y_1Y_0$.

As the sum is 3-bit long, we need 3 circuits:

\[
\begin{align*}
Y_1 & \quad X_1 & \quad Y_0 & \quad X_0 \\
\lor & \quad \lor & \quad \lor & \quad \lor \\
\lor & \quad \lor & \quad \lor & \quad \lor \\
Z_2 & \quad Z_3 & \quad Z_1 & \quad Z_0
\end{align*}
\]

Circuit Families

**Definition**

A circuit family $C$ is an infinite list of circuits, $(C_0, C_1, C_2, \ldots)$ where $C_n$ has $n$ inputs.

A circuit family is said to be uniform if there is a log-space Turing machine that on the input of $n$ 1s produces the circuit $C_n$.

For example, for computing the sum of two integers (of unlimited length), we need a circuit family:

- the even members of the family computes the sum for the least valuable segment of the numbers.
- the odd members are not needed, so they can be defined to be empty.

It is not hard to show that this family is uniform.
Parameters of Circuits

The size of a circuit is the number of gates it contains.

Two circuits are equivalent if they have the same inputs and output the same value on every input assignment.

A circuit is minimal if no smaller circuit is equivalent to it.

A circuit family is minimal if every its member is minimal.

The size complexity of a circuit family $C$ is the function $f$ on positive integers such that $f(n)$ is the size of $C_n$.

The depth of a circuit is the length of a longest path from an input to the output gate.

Depth minimal circuits and circuit families and the depth complexity of a circuit family are defined in the same way as for size.

Languages and Circuits

Definition

A circuit family $C$ decides a language $L$ over $\{0,1\}$ if, for every input string $w = a_1 a_2 \ldots a_n$

$w \in L$ if and only if $C_n$ with input $a_1 a_2 \ldots a_n$ outputs 1.

Definition

The size complexity of a language is the size complexity of a minimal circuit family that decides this language.

The depth complexity of a language is the depth complexity of a minimal circuit family that decides this language.
Example Boolean Circuit Family

Let $\mathcal{L} = \{1,11,111,1111,…\}$

We built a circuit family $C = (C_0,C_1,C_2,…)$ that decides $\mathcal{L}$

$C_n : X_1 \land X_2 \land X_3 \land … \land X_{n-1} \land X_n$

Size complexity = $n - 1$

Depth complexity = $n$

This circuit is not minimal

Size complexity of $\mathcal{L} = n - 1$

Depth complexity of $\mathcal{L} = \log n$

---

Circuit Complexity

**Theorem**

Let $f$ be a function on positive integers. Then if $\mathcal{L} \in \text{TIME}(f(n))$
then $\mathcal{L}$ has circuit complexity $O(p^2(n))$.

**Corollary**

If $\mathcal{L} \in \text{P}$, then the circuit complexity of $\mathcal{L}$ is polynomial
The Class \textbf{NC}

\textbf{Definition}

For $i \geq 1$ the class $\textbf{NC}^i$ is the class of languages that can be decided by a uniform circuit family with polynomial size complexity and depth complexity in $O(\log^i n)$

Then

$\textbf{NC} = \bigcup_{i \geq 1} \textbf{NC}^i$

\textbf{NC and Other Classes}

\textbf{Theorem}

$\textbf{NC}^1 \subseteq \textbf{L}$

\textbf{Proof Idea}

Let $\mathcal{L} \in \textbf{NC}^1$. That is there is a log-space transducer that generates a circuit family $C$ of logarithmic depth that decides $\mathcal{L}$

We have to present a log-space algorithm that decides $\mathcal{L}$

On an input $w$ of length $n$

- using the log-space transducer for $C$, generate $C_n$
- using depth-first search from the output gate check if on input $w$ the circuit outputs 1
More Theorems

**Theorem**  
NC ⊆ P

**Proof:** Obvious, since NC has polynomial size circuits

**Theorem**  
NL ⊆ NC²

**Proof:** Apply Savage’s Theorem construction, then observe that it results in a $O(\log n)^2$ depth circuit of polynomial size, so is in NC².

Brent’s Principle

(parallel time) × (number of processors) ≥ total amount of work

The total amount of work is not larger than the time complexity times the number of processors.
A Non-Parallelizable Problem

Let us consider a parallel algorithm for a NP-complete problem, say traveling salesman. Suppose there is a parallel algorithm solving this NP-complete problem.

Then there is a sequential algorithm that simulates the parallel one. By Brent’s Principle, we have

\[(\text{parallel time}) \times (\text{number of processors}) \geq \text{total amount of work}\]

where the total amount of work is not larger than the time complexity of the sequential simulation.

Either parallel time or the number of processors is exponential!!

**Bad News**

Unless \( P = \text{NP} \), no NP-complete problem can be parallelized.