Lecture 6

This lecture is on boolean circuit complexity. We first define circuits and the function they compute. Then we consider families of circuits and the language they define. In Section 2, we see how circuits relate to other complexity classes by a series of results, culminating with the Karp-Lipton theorem which states that if **NP** problems can be decided with polynomial-size circuits, then $\mathbf{PH} = \Sigma_2$.

1 Circuits

A circuit C has n inputs, m outputs, and is constructed with AND gates, OR gates and NOT gates. Each gate has in-degree 2 except the NOT gate which has in-degree 1. The out-degree can be any number. A circuit must have no cycle. See Figure 1.

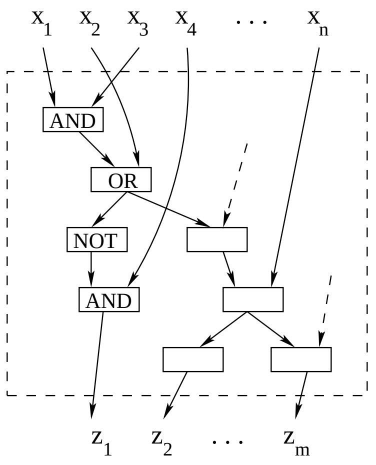

Figure 1: A Boolean circuit.

A circuit C with n inputs and m outputs computes a function $f_C : \{0,1\}^n \to \{0,1\}^m$. See Figure 2 for an example.

Define $\mathbf{SIZE}(C) = \#$ of AND and OR gates of C. By convention, we do *not* count the NOT gates.

To be compatible with other complexity classes, we need to extend the model to arbitrary input sizes:

DEFINITION 1 *A language L is solved by a family of circuits $\{C_1, C_2, \ldots, C_n, \ldots\}$ if for every $n \geq 1$ and for every x s.t. $|x| = n$,*

$$x \in L \iff f_{C_n}(x) = 1.$$

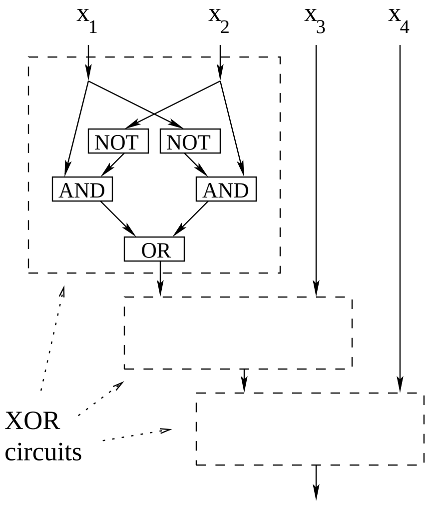

Figure 2: A circuit computing the boolean function $f_C(x_1x_2x_3x_4) = x_1 \oplus x_2 \oplus x_3 \oplus x_4$.

DEFINITION 2 *Say* $L \in \mathbf{SIZE}(s(n))$ *if* L *is solved by a family* $\{C_1, C_2, \ldots, C_n, \ldots\}$ *of circuits, where* C_i *has at most* $s(i)$ *gates.*

2 Relation to other complexity classes

PROPOSITION 1
For every language L, $L \in \mathbf{SIZE}(O(2^n))$. *In other words, exponential size circuits contain all languages.*

PROOF: We need to show that for every 1-output function $f : \{0,1\}^n \to \{0,1\}$, f has circuit size $O(2^n)$.

Use the identity $f(x_1x_2\ldots x_n) = (x_1 \wedge f(1x_2\ldots x_n)) \vee (\overline{x}_1 \wedge f(0x_2\ldots x_n))$ to recursively construct a circuit for f, as shown in Figure 3.

The recurrence relation for the size of the circuit is: $s(n) = 3 + 2s(n-1)$ with base case $s(1) = 1$, which solves to $s(n) = 2 \cdot 2^n - 3 = O(2^n)$. $\Box$

PROPOSITION 2
If $L \in \mathbf{DTIME}(t(n))$, *then* $L \in \mathbf{SIZE}(O(t^2(n)))$.

PROOF: Let L be a decision problem solved by a machine M in time $t(n)$. Fix n and x s.t. $|x| = n$, and consider the $t(n) \times t(n)$ tableau of the computation of $M(x)$. See Figure 4.

Assume that each entry (a, q) of the tableau is encoded using k bits. By Proposition 1, the transition function $\{0,1\}^{3k} \to \{0,1\}^k$ used by the machine can be implemented by a "next state circuit" of size $k \cdot O(2^k)$, which is exponential in k but constant in n. This building block can be used to create a circuit of size $O(t^2(n))$ that computes the complete tableau, thus also computes the answer to the decision problem. This is shown in Figure 5.
$\Box$

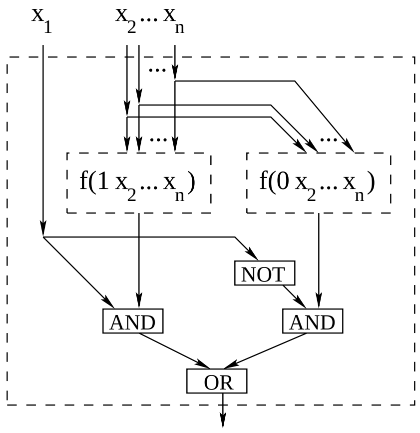

Figure 3: A circuit computing any function $f(x_1 x_2 \dots x_n)$ of n variables assuming circuits for two functions of $n-1$ variables.

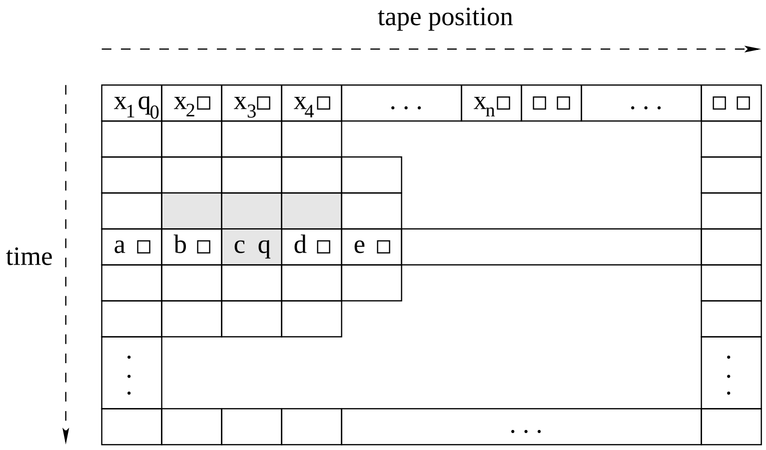

Figure 4: $t(n) \times t(n)$ tableau of computation. The left entry of each cell is the tape symbol at that position and time. The right entry is the machine state or a blank symbol, depending on the position of the machine head.

Corollary 3
$\mathbf{P} \subseteq \mathbf{SIZE}(n^{O(1)})$.

Exercise 1 $\mathbf{P} \neq \mathbf{SIZE}(n^{O(1)})$.

Proposition 4
There are languages L such that $L \in \mathbf{SIZE}(2^{o(n)})$. In other words, for every n, there exists $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by a circuit of size $2^{o(n)}$.

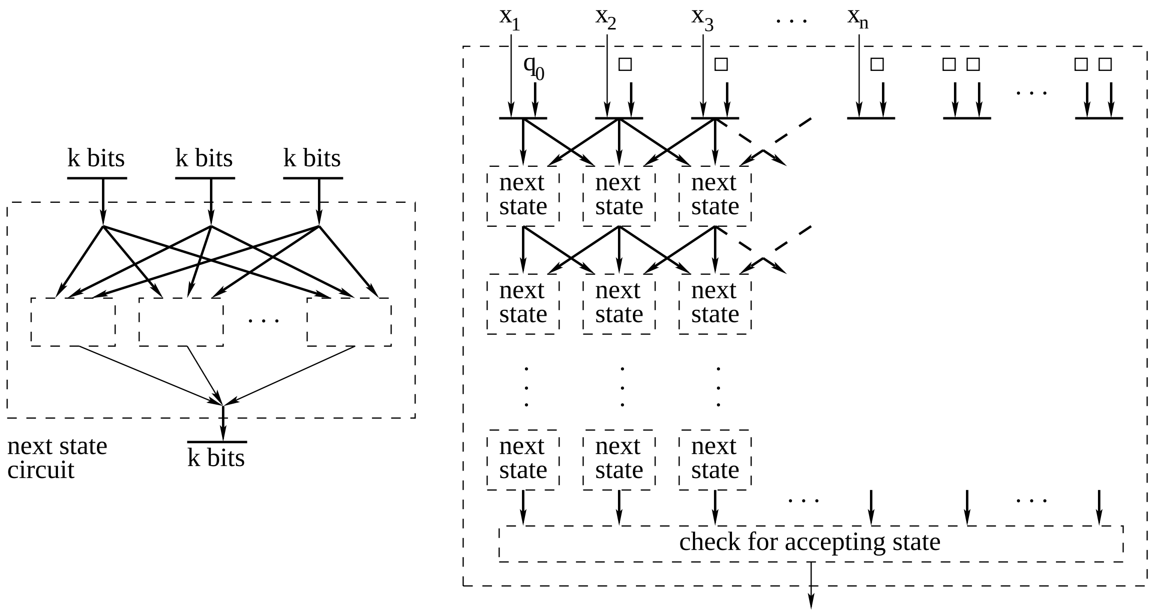

Figure 5: Circuit to simulate a Turing machine computation by constructing the tableau.

PROOF: This is a counting argument. There are 2^{2^n} functions $f : \{0,1\}^n \to \{0,1\}$, and we claim that the number of circuits of size s is at most $2^{O(s \log s)}$, assuming $s \geq n$. To bound the number of circuits of size s we create a compact binary encoding of such circuits. Identify gates with numbers $1, \ldots, s$. For each gate, specify where the two inputs are coming from, whether they are complemented, and the type of gate. The total number of bits required to represent the circuit is

$$s(2\log(n+s)+3) \leq s(2\log 2s + 3) = s(2\log 2s + 5).$$

So the number of circuits of size s is at most $2^{2s\log s + 5s}$, and this is not sufficient to compute all possible functions if

$$2^{2s\log s + 5s} < 2^{2^n}.$$

This is satisfied if $s = 2^{o(n)}$. $\Box$

EXERCISE 2 Show that there is a language in $\mathbf{SPACE}(2^{n^{O(1)}})$ that does not belong to $\mathbf{SIZE}(2^{o(n)})$.

THEOREM 5 (KARP-LIPTON)
If $\mathbf{NP} \subseteq \mathbf{SIZE}(n^{O(1)})$ *then* $\mathbf{PH} = \Sigma_2$. *In other words, the polynomial hierarchy would collapse to its second level.*

PROOF: We will show that if $\mathbf{NP} \subseteq \mathbf{SIZE}(n^{O(1)})$ then $\Pi_2 \subseteq \Sigma_2$. By a result in a previous lecture, this implies that $\mathbf{PH} = \Sigma_2$. Recall the definitions:

L is in Σ_2 if there is a polynomial time verifier V and polynomials p_1, p_2 s.t.

$$x \in L \;\Leftrightarrow\; \exists y_1\, |y_1| \leq p_1(|x|) \cdot \forall y_2\, |y_2| \leq p_2(|x|) \cdot V(x, y_1, y_2) = 1.$$

Similarly, L is in Π_2 [...]

$$x \in L \;\Leftrightarrow\; \forall y_1\, |y_1| \leq p_1(|x|) \cdot \exists y_2\, |y_2| \leq p_2(|x|) \cdot V(x, y_1, y_2) = 1.$$

Assume $\mathbf{NP} \subseteq \mathbf{SIZE}(n^{O(1)})$ and let L be in Π_2. For every n, there is a circuit C_n such that $C_n(x, y_1) = 1$ iff $\exists y_2$ such that $V(x, y_1, y_2) = 1$, where $|x| = n$, $|y_1| = p_1(n)$, $|y_2| = p_2(n)$. So,

$$\begin{aligned}
x \in L \;\Leftrightarrow\; & \exists C_n \cdot (\forall y_1 \cdot C_n(x, y_1) = 1) \\
& \text{and } C_n \text{ is the circuit that decides whether } \exists y_2 \cdot V(x, y_1, y_2) = 1 \\
\Leftrightarrow\; & \exists C_n \cdot (\forall y_1 \cdot C_n(x, y_1) = 1) \\
& \wedge\, \forall y_1 \{(\exists y_2 \cdot V(x, y_1, y_2) = 1) \Leftrightarrow C_n(x, y_1) = 1\} \quad \text{(checking specifications)}
\end{aligned}$$

... which is a Σ_3 formula, we are not in good shape. We need the circuit for the **NP** problem to not only answer the decision problem, but also to report a witness if the answer is YES.

(Second try.) There is a circuit C_n such that

$$C_n(x, y_1) = \begin{cases} 0 & \text{if there is no } y_2 \text{ such that } V(x, y_1, y_2) = 1 \\ 1y_2 & \text{otherwise, where } y_2 \text{ is such that } V(x, y_1, y_2) = 1. \end{cases}$$

See Figure 6 if you can't immediately see how a decision problem solver can always be converted into a witness finder.

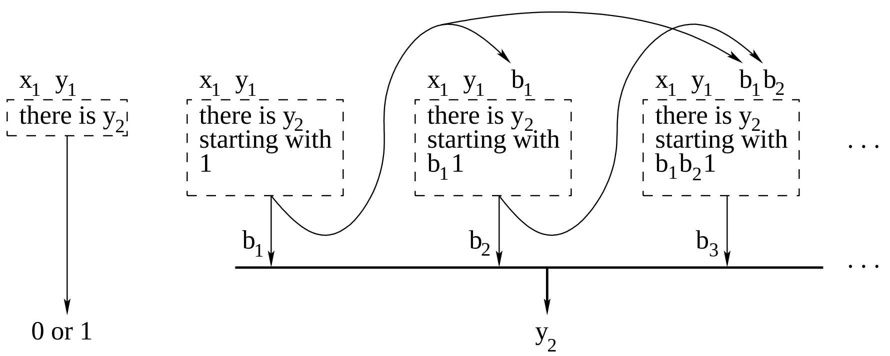

Figure 6: How to use decision problem solvers to find a witness to a search problem.

Let C be a circuit. The formula for checking specifications is

$$\begin{aligned} \forall x \forall y_1 \forall y_2 \cdot \quad & (V(x, y_1, y_2) = 1) \Rightarrow (C(x, y_1) \neq 0) \\ & \wedge\ (C(x, y_1) \neq 0) \Rightarrow (V(x, y_1, C(x, y_1)) = 1) \end{aligned}$$

which can be used to build a Σ_2 formula for L as follows:

$$\begin{aligned} x \in L \ \Leftrightarrow\ \exists C_n \cdot \forall z \forall y_1 \forall y_2 \cdot \quad & (C_n(x, y_1) \neq 0) \\ & \wedge\ (V(z, y_1, y_2) = 1) \Rightarrow (C_n(z, y_1) \neq 0) \\ & \wedge\ (C_n(z, y_1) \neq 0) \Rightarrow (V(z, y_1, C_n(z, y_1)) = 1). \end{aligned}$$

□