CSG399: Gems of Theoretical Computer Science. Lecture 10. Feb. 10, 2009.
Instructor: Emanuele Viola Scribe: Ravi Sundaram

Log-depth linear-size $\subseteq$ depth-3 subexponential-size

A major challenge in complexity theory is to exhibit an explicit function $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by fan-in 2 circuits of linear size $O(n)$ and logarithmic depth $O(\log n)$. This is sometimes referred to as the "linear-size log-depth barrier." In this lecture we prove that these circuits can be simulated by unbounded fan-in sub-exponential size circuits of depth 3. Thus, proving an exponential lower bound for depth-3 circuits would break the linear-size log-depth barrier.

At the end of the lecture we discuss the role of depth-reduction in complexity theory.

Theorem 1. *Let $C : \{0,1\}^n \to \{0,1\}$ be a circuit of size $c \cdot n$, depth $c \cdot \log n$ and fan-in 2. The function computed by C can also be computed by an unbounded fan-in circuit of size $2^{c' \cdot n/\log\log n}$ and depth-3 with inputs $x_1, x_2, \ldots, x_n, \bar{x_1}, \bar{x_2}, \ldots, \bar{x_n}$, where c' depends only on c.*

The proof of this result uses graph-theoretic techniques developed by Erdős, Graham, and Szemerédi in '75 and refined by Valiant in '77. The result is usually credited to Valiant. We are unaware of any paper where it is fully presented.

The idea of the simulation is to identify $\epsilon \cdot n$ wires to remove from C so that the resulting circuit is disconnected and each of its connected components has depth $\leq \epsilon \cdot \log n$. Each component can only depend on n^ϵ input bits, and so, given the assignment to the removed edges, can be computed in brute-force by a depth-2 circuit of sub-exponential size. Trying all $2^{\epsilon \cdot n}$ assignments to the removed edges completes the simulation. We now proceed with a formal proof. See Figure 1 for an example.

A circuit can be viewed as an acyclic directed graph with nodes representing gates and directed edges representing the flow of computed values from the output of one gate to the input of the next. The graph corresponding to C is connected, but we will also work with disconnected graphs, called forests.

Definition 2 (Depth)**.** *The* depth *of a node in a forest is the number of nodes in a longest directed path terminating at that node. The depth of the forest is the depth of a deepest node in the forest.*

Given a forest $G = (V, E)$, a depth function *D is a map $D : V \to \{1, 2, \ldots, 2^k\}$ such that if $(a, b) \in E$ then $D(a) < D(b)$.*

Claim 1. *A forest $G = (V, E)$ has depth at most 2^k if and only if there is a depth function $D : V \to \{0,1\}^k$.*

Proof. $\Rightarrow$: if the depth is at most 2^k then setting D to be the function that maps each node to its depth is a depth function.

$\Leftarrow$: suppose G has a node of depth $> 2^k$. Then there is a directed path with $> 2^k$ nodes in G. No depth function with range $\{1, 2, \ldots, 2^k\}$ can assign values to all nodes on that path. $\square$

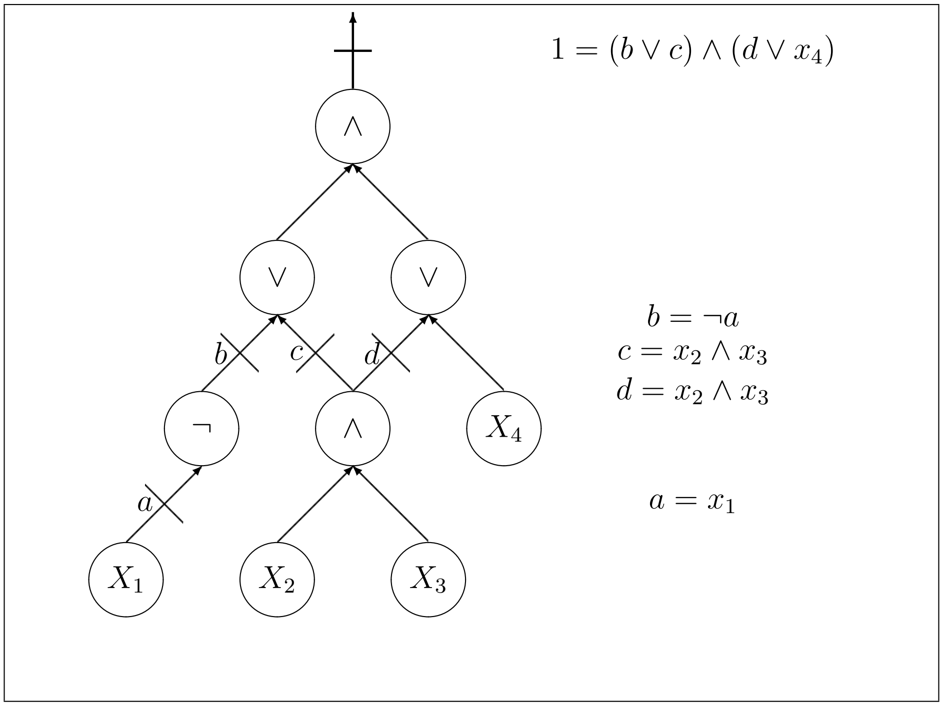

Figure 1: The removal of edges a, b, c, and d reduces the depth. The circuit evaluates to 1 if and only if there are $a, b, c, d \in \{0, 1\}$ satisfying the corresponding equations.

The following is the key lemma which allows us to reduce the depth of a forest by removing few edges.

Lemma 3. *Let $G = (V, E)$ be a forest with w edges and depth 2^k. It is possible to remove $\leq w/k$ edges so that the depth of the resulting forest is $\leq 2^{k-1}$.*

Proof. Let $D : V \to \{1, 2, \dots, 2^k\}$ be a depth function for G. Define

$E_i := \{(a, b) \in E |$ the most significant bit position where $D(a)$ and $D(b)$ differ is the i-th.$\}$

Note that $E_1, E_2, \dots, E_k$ is a partition of E. And since $|E| = w$ there exists an index $i, 1 \leq i \leq k$, such that $|E_i| \leq w/k$. We remove E_i. We now show that the depth of the resulting forest is at most 2^{k-1}. To do so we exhibit a depth function $D' : V \to \{0, 1\}^{k-1}$. Specifically, let D' be D without the i-th bit.

We claim that D' is a valid depth function for the forest $G' := (V, E \setminus E_i)$. To do this we need to show that if $(a, b) \in E \setminus E_i$ then $D'(a) < D'(b)$. Let $(a, b) \in E \setminus E_i$. Since $(a, b) \in E$, we have $D(a) < D(b)$. Now, consider the most significant bit position j where $D(a)$ and $D(b)$ differed. There are three cases to consider:

- j is more significant than i. In this case, since the j-th bit is retained, the relationship is maintained, i.e., $D'(a) < D'(b)$;

- $j = i$. This case cannot occur because the edge $(a, b) \in E_i$;
- j is less significant than i. In this case, the i-th bit of $D(a)$ and $D(b)$ is the same and so removing maintains the relationship, i.e., $D'(a) < D'(b)$.

$\square$

Now we prove the the main theorem.

Proof of Theorem 1. For simplicity, we assume that both c and $\log n$ are powers of two. Let $2^\ell := c \cdot \log n$.

Applying the above lemma we can reduce the depth by a factor 1/2, i.e. from 2^ℓ to $2^{\ell-1}$, by removing $c \cdot n/\ell$ edges. Applying the lemma again we reduce the depth to $2^{\ell-2}$ by removing $c \cdot n/(\ell - 1)$ edges. If we repeatedly apply the lemma $\log(2c)$ times the depth reduces to

$$\frac{c \log n}{2^{\log(2c)}} = \frac{\log n}{2},$$

and the total number of edges removed is at most

$$c \cdot n \left(\frac{1}{\ell} + \frac{1}{\ell - 1} + \ldots + \frac{1}{\ell - \log(2c) + 1} \right) \leq \frac{(\log 2c)c \cdot n}{\ell - \log(2c) + 1} = \frac{(\log 2c)c \cdot n}{\log \log n}.$$

For convenience we also think of removing the output edge e_{output} of the circuit. This way we can represent the output of the circuit in terms of the value of e_{output}. We define the depth of an edge $e = g \to g'$ as the depth of g, and the value of e on an input x as the value of the gate g.

For every input $x \in \{0, 1\}^n$ there exists a unique assignment h to the removed edges that corresponds to the computation of $C(x)$. Given an arbitrary assignment h and an input x we check if h is the correct assignment by verifying if the value of every removed edge $e = g \to g'$ is correctly computed from (1) the values of the removed edges whose depth is less than that of e, and (2) the values of the input bits g is connected to. Since the depth of the component is $\leq (\log n)/2$, at most $\sqrt{n}$ input bits are connected to g; we denote them by $x|_e$. Thus, for a fixed assignment h, the check for e can be implemented by a function $f_h^e : \{0, 1\}^{\sqrt{n}} \to \{0, 1\}$ (when fed the $\leq \sqrt{n}$ input bits connected to g, i.e. $x|_e$).

Induction on depth shows:

$$\begin{aligned} C(x) = 1 \Leftrightarrow \exists \text{ assignment } h \text{ to removed edges } &: \\ & h(e_{\text{output}}) = 1 \text{ and} \\ & \forall \text{ removed edge } e : f_h^e(x|_e) = 1. \end{aligned}$$

We now claim that the above expression for the computation $C(x)$ can be implemented with the desired resources. Since we removed $O(n/\log\log n)$ edges, the existential quantification over all assignments to these edges can be implemented with an OR gate with fan-in $2^{O(n/\log\log n)}$. Each function $f_h^e(x|_e)$ can be implemented via brute-force by a CNF, i.e. a depth-2 AND-OR circuit, of size $O\left(\sqrt{n}\cdot 2^{\sqrt{n}}\right)$. By collapsing the top AND gate of these AND-OR circuits with the universal quantification over all removed edges, we obtain a depth-3 circuit of size

$$2^{O(n/\log\log n)}\cdot O\left(\sqrt{n}\cdot 2^{\sqrt{n}}\right) = 2^{O(n/\log\log n)}.$$

□

Earlier we proved lower bounds for depth-d circuits of $2^{n^{\Omega(1/d)}}$. The above result shows an interesting consequence of improving such bounds to $2^{\omega(n/\log\log n)}$, even for $d = 3$. The bounds were proved for the parity function, which can be computed by depth-3 circuits of size $2^{O(\sqrt{n})}$, so one needs a different candidate.

The central role of depth reduction in complexity

Theorem 1 can be seen as one of the many depth-reduction results in complexity, where one computational model is simulated by a shallower one. Other notable examples are:

- SAT is NP-complete: To prove this central result one shows that given an NP machine with an input x, i.e. a polynomial-size non-deterministic circuit, the associated computation can be written as a SAT instance, i.e. a polynomial-size non-deterministic circuit of depth 2. Note how much power non-determinism gives! This lecture shows a simulation of linear-size log-depth circuits by *deterministic* depth-3 circuits of subexponential size.
- Toda's theorem that PH is in $P^{\#P}$: Any constant number of $\exists, \forall$ quantifications can be replaced by three quantifications: majority, majority, $\forall$.
- The lower bounds for parity: They go by showing that any constant-depth circuit can be simulated by a low-degree polynomial, which can be seen as a depth-2 circuit. (This result and Toda's share many features.)