CS 388T: Advanced Complexity Theory Spring 2020

Circuit lower bounds for PH, Karp-Lipton Theorem

Prof. Dana Moshkovitz *Scribe: Maruth Goyal*

1 Overview

Turing Machines used to be the de facto tool used to analyze relations between various complexity classes. However, when Baker, Gill, and Solovay [BGS75] introduced the Relativization barrier, it was apparent that an alternate tool was needed. This is where circuits enter the story of complexity theory. Circuits had long been studied for their applications in other fields like Electrical Engineering, and computer hardware in general. Moreover, they represent a simpler world where everything is in the form of bits, and a limited sets of gates. Thus, it becomes easier to analyze more explicit constructions, and hence bypass the relativization barrier. We will look at some proofs establishing lower bounds on the complexity of circuits needed to compute certain classes.

2 Circuits

Generally, a circuit is composed of wires, each of which may be 0 or 1. These wires feed into various "gates", including $\text{AND}(\wedge), \text{OR}(\vee), \text{NOT}, (\neg)$. Each gate has an assosciated "fan-in", i.e., the number of wires feeding into it, and "fan-out", number of wires going out. Generally unless specified otherwise we consider a fan-in of 2, and fan out of 1. However, wires may be reused as many times as one fancies.

Definition 1. *For any $T : \mathbb{N} \to \mathbb{N}$, a T-size circuit family $\{C_n\}$ is a set of circuits such that for each $n \in \mathbb{N}$ there is a boolean circuits with at most $T(n)$ gates, solving inputs of length n.*

Definition 2. *The class of languages decidable by T-size circuits is denoted as $\mathsf{SIZE}(T(n))$.*

Definition 3. *The class P/poly of languages decidable by polynomial sized circuits is defined as $\bigcup_k \mathsf{SIZE}(n^k)$.*

Observe that in our definitions above, we considered a circuit family as opposed to a single circuit. In particular, there was a different circuit for each input length n. This is a departure from the Turing Machine model we know and love, where there was just a fixed TM. This is known as the **"non-uniform"** model of computation. This model is weirdly powerful in some ways. For instance, consider any unary language. Since either each unary string is in the language or not, we just have the corresponding circuit output either 0 or 1. However, this means we can solve the unary version of the halting problem, which is otherwise undecidable.

We can reduce the power of circuits by considering a uniform restriction.

Definition 4. *A circuit family $\{C_n\}$ is said to be uniform, if there exists a Turing Machine M that on input 1^n terminates with output C_n.*

This gives us an alternate defintion of P.

Definition 5. $\mathsf{P} = \{L$ *such that there exists a TM* M *which outputs* C_n *on input* 1^n *in polynomial time, and* C_n *is polynomial size* $\}$.

Similarly, we can define P/poly in terms of P.

Definition 6. P/poly *is the set of languages decidable by polynomial time Turing Machines which are given* $\text{poly}(n)$ *bits of "advice" which depend only on* n*, but not on the input* x*.*

Observe this corresponds exactly to picking C_n for the corresponding n in the circuit family.

3 Circuit Lower Bounds

In the world of circuits, instead of working with P, we work with P/poly. Hence, instead of working with the question $\mathsf{NP} \nsubseteq \mathsf{P}$, we consider $\mathsf{NP} \nsubseteq \mathsf{P}/\text{poly}$. Observe that since $\mathsf{P} \subseteq \mathsf{P}/\text{poly}$, proving this separation also gives us $\mathsf{NP} \nsubseteq \mathsf{P}$. While we have not come close to proving this separation, other interesting circuit lower bounds in similar vein have been shown.

We first present Shannon's counting argument:

Theorem 1. (Shannon)*: Most boolean functions require* $\geq \Omega(\frac{2^n}{n})$*-size circuits.*

Proof. Observe that there are 2^{2^n} boolean functions $f : \{0,1\}^n \to \{0,1\}$, since each of the 2^n possible inputs can be given any of 2 values. Now consider any size-s circuit. i.e., it has s gates. Each "gate" can either be one of the n input bits, or $\wedge, \vee, \neg$. Thus, we have $(n+3)$ choices for it. Now, each of these s gates also has at most s^2 possible choices of incoming wires (fan-in ≤ 2). Thus, the total number of circuits of size s is

$$((n+3) \times s)^s = 2^{s(\log n + 3 + \log s)}$$

Now observe for $s = \dfrac{2^n}{10n}$ the above expression is less than 2^{2^n}.

$$2^{(2^n/(10n))(\log(n+3)+n-\log 10-\log n)} \approx 2^{2^n/10} < 2^{2^n}$$

Thus the number of functions computable by $\dfrac{2^n}{10n}$ circuits is less than the number of boolean functions of n variables. Hence, there exists functions which require more gates. Moreover, in the limit as $n \to \infty$, most functions require more gates. □

We now turn to Kannan's theorem [Kan82] which separates the polynomial hierarchy PH from $\mathsf{SIZE}(n^k)$ for all k.

Theorem 2. [Kan82] $\Sigma_3 \notin \mathsf{SIZE}(n^k)$ *for all* $k > 0$

Proof. We first consider an argument similar to Shannon's to show that there are boolean functions $f : \{0,1\}^n \to \{0,1\}$ which do not have circuits of size n^k, but moreover that all their non-zero entries

are in the first n^{k+1} rows of the truth table. Suppose all the non-zero entries are in the first n^{k+1} rows. Then clearly, since all other rows are fixed, there are $2^{n^{k+1}}$ such functions. Recall that the number of s-size circuits is about $2^{s \log s}$. Then, for sufficiently large n observe that $2^{n^{k+1}} > 2^{n^k \log n^k}$. For instance, $n \approx 2^k$ satisfies this. Thus, there are such functions which do not have n^k-size circuits. To complete the proof, we construct these circuits using a Σ_3 machine.

Recall in a Σ_3 machine, we can guess, universally check, and guess once again. We will first guess the function f. Since we are considering only the first $n+1$ entries, we only need to guess those $n+1$ bits. Second, we will verify that for every n^k-size circuit, there is some input such that the circuit outputs the wrong answer. This corresponds to a language L such that

$$x \in L \iff \exists f. \forall C. (\bigvee_{x_0} C(x_0) \neq f(x_0)) \wedge f(x) = 1$$

However, we are not done yet. At this point we have enough to confirm the existence of a hard function. However, we want to produce an actual hard function f, and thus need to single out a particular hard function. To do this, we will consider the lexically first such function by verifying that for all functions lexically before f, there is a n^k-size circuit for it. Note, this is also why we impose $f(x) = 1$, to identify a single function for each x if it exists. Thus, the final expression is

$$x \in L \iff \exists f.\ \forall C.\ (\bigvee_{x_0} C(x_0) \neq f(x_0)) \wedge f(x) = 1 \quad \forall (f' < f)\ \exists C' \bigwedge_{y_0} C'(y_0) = f'(y_0)$$

Notice that the quantifiers correspond exactly to a Σ_3 sequence. Thus L defines a Σ_3 language, which by construction outputs a function f which can be evaluated in Σ_3, which does not have n^k circuits. Thus we have that $\Sigma_3 \nsubseteq \mathsf{SIZE}(n^k)$ for all $k > 0$. □

4 Karp-Lipton Theorem

We now present a theorem of Karp and Lipton [KL80]. As we mentioned above, we consider the question of $\mathsf{NP} \nsubseteq \mathsf{P/poly}$. Karp and Lipton showed the following theorem:

Theorem 3. *if* $\mathsf{NP} \subseteq \mathsf{P/poly}$ *then* $\Sigma_2 = \Pi_2$*, and hence* $\mathsf{PH} = \Sigma_2$.

That is, the entire polynomial hierarchy collapses to the second level if NP has poly-size circuits. It is very strongly believed by the complexity theory community that the polynomial hierarchy does not indeed collapse, at least to such a low level, and so we believe that $\mathsf{NP} \nsubseteq \mathsf{P/poly}$. The proof of this theorem is so short it has been featured on tote bags at conferences. Ryan Williams even suggested reciting it as a measurement for how long one must wash their hands.

Proof. Any Π_2 language is of the form $x \in L \iff \forall y.\ \exists z.\ \varphi(x, y, z) = 1$. If $\mathsf{NP} \in \mathsf{P/poly}$ then in particular $3 - \mathsf{SAT}$ has poly-size circuits. From this we can generate a poly-size circuit that outputs a satisfying assignment if it exists, not just whether a circuit is satisfiable. Given this, we can simulate Π_2 in Σ_2. We simply guess this poly-size assignment-generating circuit, i.e. $x \in L \iff \exists C.\ \forall y.\ \varphi(x, y, C(\varphi, x, y)) = 1$. Thus $\Sigma_2 = \Pi_2$. But then observe that any alternation of quantifiers can be reduced down to $\exists\forall$. eg:

$$\exists\forall\exists\forall \to \exists\forall\exists\exists\forall \to \exists\forall\exists\forall \to \exists\forall\forall\exists \to \exists\forall\exists \to \exists\exists\forall \to \exists\forall$$

Thus, $\mathsf{PH} = \Sigma_2$. □

From this, we can also conclude the following

Theorem 4. $\Sigma_2 \nsubseteq \mathsf{SIZE}(n^k)$

Proof. Consider 2 cases:

1. $\mathsf{NP} \subseteq \mathsf{P/poly}$. By the Karp-Lipton theorem, $\mathsf{PH} = \Sigma_2$, but $\mathsf{PH} \nsubseteq \mathsf{SIZE}(n^k)$ by Kannan's theorem.
2. $\mathsf{NP} \nsubseteq \mathsf{P/poly}$. Well, $\mathsf{NP} \subseteq \Sigma_2$, so $\Sigma_2 \nsubseteq \mathsf{SIZE}(n^k)$.

□

References

[BGS75] Theodore Baker, John Gill, and Robert Solovay. Relativizations of the p=?np question. *SIAM Journal on computing*, 4(4):431–442, 1975.

[Kan82] Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and control*, 55(1-3):40–56, 1982.

[KL80] Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, STOC '80, page 302–309, New York, NY, USA, 1980. Association for Computing Machinery.