Computability and Complexity

Lecture 8, Thursday August 24, 2023

Ari Feiglin

8.1 Nonuniform Computation

Definition 8.1:

We say that a decision problem S is **decided** by a sequence of boolean circuits $\{C_n\}_{n=0}^{\infty}$ where for every $x \in \{0,1\}^*$,

$$x \in S \iff C_{|x|}(x) = 1$$

So C_n is the boolean circuit which decides if an input of size n is in S.

The **complexity** of a boolean circuit is defined to be the number of edges in it. The complexity of a boolean circuit C is denoted by $|C|$.

We say that a decision problem S is **decidable by a sequence of polynomial circuits** if there exists a sequence of boolean circuits $\{C_n\}_{n=0}^{\infty}$ which decides S, and a polynomial p such that for every n,

$$|C_n| \leq p(n)$$

Definition 8.2:

Given a function $\ell\colon \mathbb{N} \longrightarrow \mathbb{N}$, we define the class of decision problems $\mathbf{P}/_{\ell}$. A decision problem S is in $\mathbf{P}/_{\ell}$ if and only if there exists a polynomial-time algorithm A and a sequence of commands $\{a_n\}_{n=0}^{\infty}$ where

(1) For every n, $|a_n| \leq \ell(n)$

(2) For every x, x is in S if and only if $A(a_{|x|}, x) = 1$.

We define the class

$$\mathbf{P}/_{\mathsf{poly}} = \bigcup_{\substack{\ell \text{ is a} \\ \text{polynomial}}} \mathbf{P}/_{\ell} = \bigcup_{k=0}^{\infty} \mathbf{P}/_{n^k}$$

Proposition 8.3:

A decision problem S is in $\mathbf{P}/_{\mathsf{poly}}$ if and only if S can be decided by a sequence of polynomial circuits.

Proof:

Suppose S can be decided by a sequence of polynomial circuits $\{C_n\}_{n=0}^{\infty}$ whose complexity is bound by a polynomial p. We define the algorithm $A(C, x)$ which takes as input a boolean circuit C and runs it with the input x. Then since $|C_n| \leq p(n)$ and for every x, $x \in S$ if and only if $C_{|x|}(x) = 1$ which is if and only if $A(C_{|x|}, x) = 1$. So

$$S \in \mathbf{P}/_{p} \subseteq \mathbf{P}/_{\mathsf{poly}}$$

Now suppose that S is in $\mathbf{P}/_{\mathsf{poly}}$, so there exists a polynomial p and a sequence of commands $\{a_n\}_{n=0}^{\infty}$ such that $|a_n| \leq p(n)$ and $x \in S$ if and only if $A(a_{|x|}, x) = 1$. We showed in our proof that CSAT is $\mathbf{NP}$-complete, that for every polynomial algorithm there exists a polynomial circuit which computes the algorithm. So there exists a boolean circuit C_n which is equivalent to computing A which has $n + |a_n|$ input nodes, and whose size is polynomial in $n + |a_n|$. Let us plug in a_n to the input nodes corresponding to it (the first input nodes) in C_n and this gives us a boolean circuit C'_n where $C'_n(x) = A(a_n, x)$. And the complexity of C_n is polynomial in n, so $\{C_n\}_{n=0}^{\infty}$ is a sequence of polynomial circuits which decides S. ■

Notice that $\mathbf{P}/_0$ is equal to $\mathbf{P}$, since for $S \in \mathbf{P}/_0$, the size of each command a_n is zero, so they are all empty. Thus we can define a polynomial-time algorithm $B(x) = A(\varepsilon, x)$ and so $x \in S$ if and only if $A(a_{|x|}, x) = A(\varepsilon, x) = B(x) = 1$. So B solves S, meaning $S \in \mathbf{P}$. And if $S \in \mathbf{P}$, it is solved by $B(x)$, so define $A(a, x) = B(x)$ and $a_n = \varepsilon$. So

$$\mathbf{P}/_0 = \mathbf{P}$$

Proposition 8.4:

$\mathbf{P}/_1$ contains undecidable problems.

Proof:

Let S be an undecidable problem which is encoded in unary, ie. $S \subseteq \{1\}^*$ is undecidable. Let us define $S' = \left\{x \mid 1^{|x|} \in S\right\}$, then S' is also undecidable as otherwise S would be (define an algorithm where given a unary encoding 1^n, it checks for some x whose length is n if $x \in S'$). Now we will prove $S' \in \mathbf{P}/_1$. Let us define

$$a_n = \begin{cases} 1 & 1^n \in S \\ 0 & \text{else} \end{cases}$$

And we define $A(a_{|x|}, x) = a_{|x|}$ (ie. $A(a, x) = a$). Then A is polynomial-time and solves S', as $A(a_{|x|}, x) = 1$ if and only if $|x| \in S$, which is if and only if $x \in S'$. So $S' \in \mathbf{P}/_1$ despite S' being undecidable. ■

So $\mathbf{P}/_1 \subseteq \mathbf{P}/_{\mathsf{poly}}$ contains undecidable problems, and therefore

Proposition 8.5:

$$\mathbf{P}/_{\mathsf{poly}} \not\subseteq \mathbf{NP}$$

Proposition 8.6:

For every decision problem $S \in \{0,1\}^*$, $S \in \mathbf{P}/_{2^n}$. In other words $\mathbf{P}/_{2^n} = \mathbf{P}$.

Proof:

For every n, there are 2^n binary strings of length n, and let us enumerate the binary strings of length n lexicographically by π_n (ie. $\pi_n(i)$ gives the binary string ω which is the ith smallest in the order). Notice that all π_n does is convert i to binary. π_n and its inverse π_n^{-1} can be computed in polynomial time. Then let us define a_n to be a string of length 2^n where $[a_n]_i$ is one if and only if $\pi_n(i)$ is in S. Then let $A(a_{|x|}, x)$ look at $a_{|x|}$ at index $\pi_{|x|}^{-1}(x)$ (meaning at index x, when we view x as binary) and return the value found.

Then $A(a_{|x|}, x)$ is one if and only if $\pi_{|x|}(\pi_{|x|}^{-1}(x)) = x \in S$. So A decides S, meaning $S \in \mathbf{P}/_{2^n}$. ■

Definition 8.7:

A set S is called **sparse** if there exists a polynomial p such that for every n, the number of binary strings in S of length n is bound by $p(n)$. In other words, for every n

$$|S \cap \{0,1\}^n| \leq p(n)$$

We will denote $S \cap \{0,1\}^n$ by $S^{=n}$. We define

$$\mathsf{Sparse} = \{S \mid S \text{ is a sparse set}\}$$

Notice that $\mathsf{Sparse} \subseteq \mathbf{P}/_{\mathsf{poly}}$. If $S \in \mathsf{Sparse}$ then let p be a polynomial bound of $S^{=n}$. Then for every n, we define a_n to be the list of all binary strings of length n which are in S. Since there are $\leq p(n)$ such strings, and each has a length of n, $|a_n| \leq n \cdot p(n)$ so a_n's length is bound by a polynomial. And we can define $A(a_{|x|}, x)$ to just look at the list $a_{|x|}$ and check if x is in it. This solves S, and so $S \in \mathbf{P}/_{\mathsf{poly}}$.

Proposition 8.8:

A decision problem S is in $\mathbf{P}/_{\mathsf{poly}}$ if and only if there exists a Cook reduction from S to a sparse set. In other words,

$$\mathbf{P}/_{\mathsf{poly}} = \mathbf{P}^{\mathsf{Sparse}}$$

Proof:

Suppose S has a Cook reduction to some $S' \in \mathsf{Sparse}$, let $A^{S'}$ be this Cook reduction. Suppose that $S'^{=n}$'s size is bound by $p(n)$. Let $t(n)$ be the polynomial time bound on the runtime of $A^{S'}$. Notice then that for inputs of length n, the length of $A^{S'}$'s longest possible query is bound by $t(n)$.

So for every n, we define the command a_n to be the list of all the strings in S' whose length is bound by $t(n)$. Notice then that

$$|a_n| = \sum_{i=0}^{t(n)} \left|S'^{=i}\right| \cdot i \leq \sum_{i=0}^{t(n)} i \cdot p(i)$$

We can assume that p is increasing (we can take $p = n^k$ for example), and so

$$|a_n| \leq t(n) \cdot p(t(n))$$

so a_n's length is bound by a polynomial. Now we define $A(a_{|x|}, x)$ which runs $A^{S'}(x)$ but instead of querying S''s oracle, it instead checks the list $a_{|x|}$ to see if the query is in there. This solves S in polynomial time, since $A^{S'}$ does, and since the commands are polynomially bound, we have that $S \in \mathbf{P}/_{\mathsf{poly}}$.

For the converse, suppose $S \in \mathbf{P}/_{\mathsf{poly}}$ so there exists a polynomial-time algorithm A and a polynomial p which bounds the length of the commands $\{a_n\}_{n=0}^{\infty}$. We can assume without loss of generality that $|a_n| = p(n)$, and that $p(n)$ is increasing.

For every n and for every $1 \leq i \leq p(n)$, we define a_n^i to be the word of length $p(n)$ where the ith index is 1, and all other indexes are zero. Let

$$S' = \left\{a_n^i \;\middle|\; n \in \mathbb{N},\, a_n[i] = 1\right\}$$

Notice that for every n, $|S'^{=n}| \leq n$ since there are only n choices for where to put the bit which is equal to one in a string of zeroes. So $S' \in \mathsf{Sparse}$.

Now we define $B^{S'}$:

1. **function** $B^{S'}(x)$
2. **for** $(1 \leq i \leq p(|x|))$
3. **if** $(a_{|x|}^i \in S')$ $\sigma_i \leftarrow 1$
4. **else** $\sigma_i \leftarrow 0$
5. **end for**
6. $a \leftarrow \sigma_1\sigma_2 \cdots \sigma_{p(|x|)}$
7. **return** $A(a, x)$
8. **end function**

Notice that at line 6, $a = a_{|x|}$ as $a[i]$ is one if and only if $a_{|x|}^i \in S$ which is if and only if $a_{|x|}[i]$ is one, so $a = a_{|x|}$. So $B^{S'}$ is an oracle machine which solves S in polynomial time, and thus a Cook reduction from S to S'. And since $S' \in \mathsf{Sparse}$, we have that $S \in \mathbf{P}^{\mathsf{Sparse}}$.

So we have shown inclusion in both direction, meaning $\mathbf{P}/_{\mathsf{poly}} = \mathbf{P}^{\mathsf{Sparse}}$. ■

Theorem 8.9:

If $\mathbf{NP} \subseteq \mathbf{P}/_{\mathsf{poly}}$ then $\mathbf{PH} = \Sigma_2$.

Proof:

We will show that $\Pi_2 \subseteq \Sigma_2$, and this would mean $\Sigma_2 = \Sigma_3$ which would mean $\mathbf{PH} = \Sigma_2$. So let $S \in \Pi_2$, and so there exists a polynomial time verifier V and a polynomial p such that

$$x \in S \iff \forall y \exists z \big(V(x, y, z) = 1\big)$$

where $|y|, |z| \leq p(|x|)$. Let us define

$$S' = \big\{(x,y) \;\big|\; |y| \leq p(|x|), \exists z, |z| \leq p(|x|)\big(V(x,y,z) = 1\big)\big\}$$

So

$$x \in S \iff \forall y, |y| \leq p(|x|)\big((x,y) \in S'\big)$$

And S' is in **NP** as we can define a verifier $V'\big((x,y),z\big) = V(x,y,z)$, and this is a polynomial proof system (as there exists a witness $|z| \leq p(|x|)$). Since SAT is **NP**-complete, there exists a Karp reduction f from S' to SAT. And so

$$x \in S \iff \forall y\big(f(x,y) \in \mathsf{SAT}\big)$$

with the usual condition that $|y| \leq p(|x|)$. Since $\mathsf{SAT} \in \mathbf{NP} \subseteq {}^{\mathbf{P}}\!/_{\mathsf{poly}}$, and so there exists a polynomial-time algorithm A, a polynomial q, and a sequence of commands $\{a_n\}_{n=0}^{\infty}$ whose lengths are bound by q which satisfy the condition for SAT to be in ${}^{\mathbf{P}}\!/_{\mathsf{poly}}$.

Let p_f be the time complexity for computing the Karp reduction f. So $|f(x,y)| \leq p_f(|x| + |y|)$. Since the Karp reduction f runs in $p_f(|x| + |y|)$ time and $|y| \leq p(|x|)$, we have that f runs in $p_f(n + p(n))$ time. So if $\varphi = f(x,y)$ then $|\varphi| \leq p_f(n + p(n))$, thus let us define

$$b_n = a_1 a_2 \cdots a_{p_f(n+p(n))}$$

And this means that b_n will contain $a_{|\varphi|}$. And notice that

$$|b_n| = \sum_{i=1}^{p_f(n+p(n))} q(i)i \leq p_f(n + p(n))^2 q(p_f(n + p(n)))$$

So the length of b_n is polynomial in n.

Let us define $V'(x, b_{|x|}, y)$ which functions as follows:

(1) First we compute the image of x, y in the Karp reduction, ie. let $\varphi = f(x,y)$.

(2) Since $b_{|x|}$ contains $a_{|\varphi|}$, let us run $A(a_{|\varphi|}, \varphi)$. If this returns zero, return zero since then $\varphi \notin \mathsf{SAT}$ so $(x,y) \notin S'$.

(3) Now, using A and $a_1, \ldots, a_{|\varphi|}$, we can compute a boolean vector τ which satisfies φ. This can be done by setting x_1 to true, and checking if the new boolean formula φ', whose length is less than φ, is in SAT. This can be done recursively, and so we construct τ.

Since $b_{|x|}$ isn't necessarily actually the list of commands $a_1, \ldots$, we must verify that τ actyually satisfies φ. So return if $\varphi(\tau) = 1$.

V' runs in polynomial time, since every step runs in polynomial time. And if $x \in S$ then for every $|y| \leq p(|x|)$, $f(x,y) \in \mathsf{SAT}$. And so for $b_{|x|}$ defined as the concatenation of $a_1, \ldots, a_{p_f(n+p(n))}$, $V'(x, b_{|x|}, y) = 1$. Since $b_{|x|}$'s length is polynomial in $|x|$, this means that if $x \in S$ then there exists a $b_{|x|}$ such that for all y, $V'(x,b,y) = 1$ (with the strings being polynomial in length).

If $x \notin S$ then there exists a $|y| \leq p(|x|)$ such that $f(x,y) \notin \mathsf{SAT}$ and so for this y, V' will always return zero (since the constructed τ will not satisfy φ). So if $x \notin S$, for every b there exists a y (which is this y, it is not dependent on b), such that $V'(x,b,y) = 0$. So

$$x \in S \iff \exists b \forall y\big(V(x,b,y) = 1\big)$$

And this means that $S \in \Sigma_2$, so $\Pi_2 \subseteq \Sigma_2$ as required. ■

Thus many researchers hypothesize that $\mathbf{NP} \not\subseteq {}^{\mathbf{P}}\!/_{\mathsf{poly}}$, as otherwise the polynomial hierarchy would collapse to Σ_2, and many researchers do not believe this is true.