MCS 541 Project

Gregoire Fournier

March 3, 2025

Function and TM intro.

1 Ladner's theorem: Existence of NP intermediate problems [1] [2]

Recall:

- $TIME(f(n))$ is the set of all functions computable in $O(f(n))$, or the set of all languages decided by a DTM in $O(f(n))$ time. $P = \bigcup_k TIME(n^k)$.
- $L \in NP$ **iff** $\exists p$ polynomial and $\exists M$ poly-time DTM st. $\forall x \in L$, $\exists c$, $|c| = p(x)$ with $M(x, c) = 1$.
- $NP = \bigcup_k NTIME(n^k)$ (non-deterministic choices made by an accepting computation of an NDTM can be viewed as a certificate that the input is in the language).
- A is NP-complete **iff** $A \in NP$ and $\forall L \in NP$, $L <_P A$.

A lot of NP-problems are complete.

- SAT by Cook Levin theorem.
- 3SAT, CLIQUE, VERTEX-COVER, HAMPATH, SUBSET-SUM ...

Theorem 1 (**Ladner**) If $\mathbf{P} \neq \mathbf{NP}$, then there is L st: $\begin{cases} L \in (\mathbf{NP} \setminus \mathbf{P}) \\ L \notin \mathbf{NP}\text{-complete} \end{cases}$.

Proof. Clearly $SAT \in (\mathbf{NP} \setminus \mathbf{P})$ (otherwise $\mathbf{P} = \mathbf{NP}$), for $H : \mathbb{N} \to \mathbb{N}$ poly-time computable, define SAT_H as follows:

$$SAT_H = \{w01^{n^{H(n)}} | w \in SAT, |w| = n\}$$

Then there are two main behaviours for SAT_H relying on H:

i. If H bounded, SAT_H is SAT with some polynomial size padding, so SAT_H **NP**-complete.

ii. Otherwise H unbounded, then the padding is not of polynomial size, and SAT_H cannot be **NP**-complete:

   - Otherwise $SAT <_P SAT_H$, so $\exists k$ st. there is a $O(n^k)$ reduction from SAT to SAT_H.

- An element of size n of SAT would be mapped to an element of size $O(n^k)$, but since H is unbounded, then any instance of SAT, w, must be of size $o(n)$ so that $|w|^{|w|^{H(|w|)}} = O(n^k)$ can be satisfied.
- Such a reduction poly-time reduces instances of SAT of size n to $o(n)$, so $SAT \in \mathbf{P}$ which leads to $\mathbf{P} = \mathbf{NP}$, contradiction.

Now consider such a function H:

$$H(n) := \text{smallest } i < log(logn) \text{ st. } \forall x \in \{0,1\}^*, |x| < log(n),$$

$$M_i \text{ halts on } x \text{ within } i|x|^i \text{ steps and } M_i \text{ outputs 1 iff } x \in SAT_H.$$

$$H(n) = min\{i < log(log(n)) | \forall w, |w| < log(n), M_i \text{ decides } w \in SAT_H \text{ within } i|w|^i \text{ steps}\}$$

Claim 2 (SAT_H not in $\mathbf{P}$) Suppose otherwise, so there is a TM M that decides it in $O(n^k)$ and in particular there is $i > k$ st. $M = M_i$ one of the machines described as above. So H is bounded by i, therefore by (i) SAT_H cannot be in $\mathbf{P}$ (otherwise $SAT \in \mathbf{P}$ and $\mathbf{P} = \mathbf{NP}$).

Claim 3 (SAT_H not $\mathbf{NP}$-complete) $SAT_H \notin \mathbf{P}$, so $\forall i \in \mathbb{N}$ there is $w \in SAT_H$ st. M_i takes $O(n^i)$ time to decide, so $H(n)$ is unbounded. By (ii) SAT_H not $\mathbf{NP}$-complete.

■

2 The Gap and Speed up theorems [3]

2.a Gap theorem

Recall:

- definition function space constructible: $f : \mathbb{N} \to \mathbb{N}$ is spc if there exists an $O(f(n))$ space TM exists that always halts with the bin. representation of $f(n)$ with input 1^n.
- SPACE separation/hierarchy theorem: if $f(n)$ space constructible, then $\exists L$ language decidable in $O(f(n))$ space but not in $o(f(n))$ space. (uses diagonalisation argument)

Theorem 4 (**Gap Theorem**) For any total computable function $f : \mathbb{N} \to \mathbb{N}$ st. $f(x) \geq x$, there exists a time bound $T(n)$ st $TIME(f(T(n))) = TIME(T(n))$.

Proof. Consider $T_i(x)$ the running time of TM M_i on input x. Now define:

$$\forall n, T(n) = min_{m \in \mathbb{N}}\{m | \forall i \leq n, T_i(n) \leq f(m) \to T_i(n) \leq m\}$$

To compute $T(n)$, start at $m = 0$ and if $i \leq n$ st. $m < T_i(n) \leq f(m)$, set $m := T_i(n)$. The value of $T(n)$ is the final value of the m of the process.

Claim 5 ($T(n)$ witnesses the theorem) Suppose M_i runs in time $f(T(n))$, then $T_i(n) \leq f(T(n))$ a.e. By construction for $n \geq i$, $T_i(n) \leq T(n)$.

■

2.b Speed-up theorem

Theorem 6 (**Speed-up Theorem**) For any total computable **monotone** function $f : \mathbb{N} \to \mathbb{N}$ st. $f(x) \geq x^2$, there exists a recursive set A such that for any TM M_i accepting A, there exists a TM M_j accepting A with $f(Tj(x)) < Ti(x)$ a.e.

Proof. Denoting $f^n = \underbrace{f \circ ... \circ f}_{n \text{ times}}$, construct a set $A \subseteq 0*$ with the following properties:

i. for any machine M_i accepting A, $T_i(0^n) > f^{n-i}(2)$ a.e.

ii. $\forall k$, there exists a machine M_j accepting A st. $T_j(0^n) \leq f^{n-k}(2)$ a.e.

And then :

$$\exists M_j \text{ st. } f(T_j(0^n)) \leq f(f^{n-i-1}(2)) = f^{n-i}(2) \text{ a.e. by (ii) + monotonicity of f}$$

So applying (i) yields $\exists M_j$ st. $f(T_j(0^n)) < T_i(0^n)$, which is the theorem.

Construction of the set A satisfying (i) and (ii):

Consider $M_0, M_1, ...$ TM for the alphabet $\{0\}$. Let N be an enumeration machine that maintains a finite list of descriptions of machines currently being simulated. It is assumed a description of M_i suitable for simulation is obtained from the index i.

The computation of N proceeds in stages. Initially, the list is empty. At stage n, N puts the next machine M_n at the end of the list. It then simulates the machines on the list in order.

For each such M_i, it simulates M_i on input 0^n for $f^{n-i}(2)$ steps, picks the first one that halts and return the opposite of what the machine decided. If no machine halts within the time, then N yields $0^n \notin A$.

Claim 7 (A has property (i))
The machine M_i is put on the list at stage i. If M_i halts within time $f^{n-i}(2)$ on 0^n and is not chosen for deletion, then some higher priority machine on the list must have been chosen; this can happen only finitely many times, so eventually M_i will be chosen for deletion at worse at stage n.

At that point, 0^n will be put into A iff $0^n \notin L(M_i)$, so that $L(M_i) \neq A$. Therefore any machine M_i that runs in time $f^{n-i}(2)$ i.o. does not accept A, which is (i).

Claim 8 (A has property (ii))
NTS that $\forall k$, A is accepted by a TM N_k running in time $f^{n-k}(2)$ a.e. The key idea is to hard-code the first m stages of the computation of N in the finite control of N_k, as for each M_i, either:

(a) $T_i(0^n) \leq f^{n-i}(2)$ i.o., and there is a stage $m(i)$ at which N deletes M_i from the list.

(b) $T_i(0^n) > f^{n-i}(2)$ a.e., and there is a stage $m(i)$ after which M_i always exceeds its time.

Let $m = max_{i \leq k} m(i)$. $m(i)$ and m are well defined but hard to find.

- For inputs $0^n, (n \leq m)$, N_k has a finite list of elements hard-coded, so looks up to decide $0^n \in A$.
- For $0^n, n > m$:
  - N_k simulates the action of N on stages $m+1, m+2, ..., n$ starting with a certain list hard-coded in its finite control. The list it starts with is the list of N at stage m with all $i \leq k$ machines M_i deleted.

    The simulation behave as N would at stage m and beyond: $\forall i, M_i$ with $i \leq k$, either :

    (a) it has been deleted from the list by stage m or

    (b) it will always exceed its allotted time after stage m, so will not be deleted

    N_k can thus decide $0^n \in A$.
  - It remains to estimate the running time of N_k on input 0^n:
    - If $n \leq m$, N_k takes linear time (read + lookup).
    - If $n > m$, N_k simulates at most $n-k$ machines of the list on $n-m$ inputs, each for at most $f^{n-k-1}(2)$ steps. M_i has at most $log(i)$ states, at most $logi$ tape symbols, at most $logi$ transitions in its finite control and so one step of M_i can be simulated in $c(log(i))^2$ steps of N_k.
      Thus the total time for the simulations is at most $cn^2(logn)^2 f^{n-k-1}(2)$ and:

      $$cn^2(logn)^2 f^{n-k-1}(2) \leq 2^{2^{n-k-1}} \leq f^{n-k-1}(2), \text{ because } f(m) \geq m^2.$$

      So:

      $$cn^2(logn)^2 f^{n-k-1}(2) \leq f^{n-k-1}(2)^2 \leq f(f^{n-k-1}(2)) = f^{n-k}(2), \text{ (ii)}.$$

■

3 The Arithmetic hierarchy [3][4]

3.a Definitions

Let A, B be sets of strings.

- A is recursively enumerable in B if $A = L(M^B)$ for some oracle TM M with oracle B
- A is recursive in B if $A = L(M^B)$ for some oracle TM M with oracle B such that M^B is total (i.e membership in A is decidable relative to an oracle for B), and write $A \leq_T B$. The relation $\leq_T$ is called Turing reducibility.

Then define a hierarchy of classes above the r.e. sets (analogous to the polynomial time hierarchy) as follows. Fix the alphabet $\{0, 1\}$ and identify strings in $\{0, 1\}$ with the natural numbers.

$$\Sigma_1^0 := \{\text{r.e. sets}\}$$

$$\Delta_1^0 := \{\text{recursive sets}\}$$

$$\Sigma_{n+1}^0 := \{L(M^B) | B \in \Sigma_n^0\} = \{A | A \text{ is r.e. in some } B \in \Sigma_n^0\}$$

$$\Delta_{n+1}^0 := \{L(M^B) | B \in \Sigma_n^0, M^B total\} = \{A | A \text{ is recursive in some } B \in \Sigma_n^0\}$$

$$\Pi_n^0 := \{\text{complements of sets in } \Sigma_n^0\}$$

3.b Characterisation in terms of quantifiers and reduction

Theorem 9 i. A set A is in Σ_n^0 iff there exists a decidable $(n+1)$-ary predicate R st:

$$A = \{x | \exists y_1 \forall y_2 \exists y_3 ... Q y_n R(x, y_1, ..., y_n)\} \text{ where } Q \text{ is } \begin{cases} \exists \text{ if } n \text{ is odd} \\ \forall \text{ otherwise} \end{cases}$$

ii. A set A is in Π_n^0 iff there exists a decidable $(n+1)$-ary predicate R st:

$$A = \{x | \forall y_1 \exists y_2 \forall y_3 ... Q y_n R(x, y_1, ..., y_n)\} \text{ where } Q \text{ is } \begin{cases} \forall \text{ if } n \text{ is odd} \\ \exists \text{ otherwise} \end{cases}$$

iii. $\Delta_n^0 = \Sigma_n^0 \cap \Pi_n^0$

For $A \subseteq \Sigma*$ and $B \subseteq \Gamma*$, define $A \leq_m B$ (A many-one reducible to B) if there exists a total recursive function $\sigma : \Sigma \to \Gamma$ st:

$$\forall x \in \Sigma,\ x \in A \text{ iff } \sigma(x) \in B$$

3.c Examples

3.c.1 HP is $\leq_m$-complete for Σ_1^0

$$HP = \{(M, x) | \exists t \text{ st } M \text{ halts on } x \text{ in } t \text{ steps } \}$$

For any TM M, the map $x \to M\#x$ is a total computable map reducing $L(M)$ to HP.

3.c.2 MP is $\leq_m$-complete for Σ_1^0

$$MP = \{(M, x) | \exists t \text{ st } M \text{ accepts } x \text{ in } t \text{ steps } \}$$

For any TM M, the map $x \to M\#x$ is a total computable map reducing $L(M)$ to MP.

3.c.3 EMPTY is $\leq_m$-complete forΠ_1^0

3.c.4 TOTAL is $\leq_m$-complete for Π_2^0

3.c.5 FIN is $\leq_m$-complete for Σ_2^0

3.c.6 COF is $\leq_m$-complete forΣ_3^0

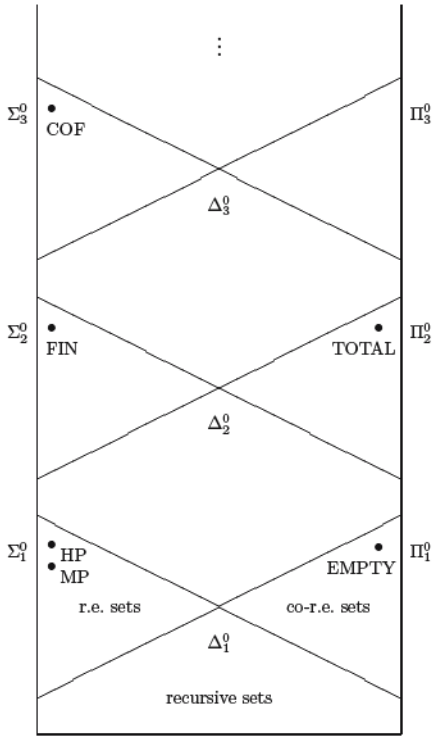

Figure 1: The arithmetic hierarchy [3]

3.c.7 If A, B r.e, so are $A \cup B$ and $A \cap B$.

- $A \cup B$: By proposition 8.2[4], there are f, g total recursive such that $A = ran(f)$ and $B = ran(g)$. Define h as follows :
  $\forall x \in \mathbb{N}, h(2x) = f(x)$ and $h(2x+1) = g(x)$. h is total recursive and $ran(h) = ran(f) \cup ran(g) = A \cup B$. So by 8.2[4], $A \cup B$ is r.e.

- $A \cap B$: By definition, there are f, g partial recursive such that $A = dom(f)$ and $B = dom(g)$. Define h as follows :
  $\forall x \in \mathbb{N}, h(x) = f(x) + g(x)$. h is partial recursive and $dom(h) = dom(f) \cap dom(g) = A \cap B$. So by definition, $A \cap B$ is r.e.

3.c.8 Let $(W_e)_{e \in \mathbb{N}}$ enumeration for the r.e subsets of $\mathbb{N}$

3.c.8.1 A is Σ_3^0

$$A = \{e : \exists h \forall n (2^h + n2^{h+1} \in W_e)\}$$

Which is the same as :

$$A = \{e : \exists h \forall n \exists s (2^h + n2^{h+1} \in W_e^s)\}$$

Since W_e^s is a Δ_1 set, by proposition 8.15[4] A is Σ_3^0.

3.c.8.2 Every Σ_3^0 subset of $\mathbb{N}$ is many-one reducible to A

Let B be any Σ_3^0 set. Then there is a recursive function $R(x, h, n, s)$ such that :

$$x \in B \Leftrightarrow \exists h \forall n \exists s R(x, h, n, s)$$

Let $f(x,n) = \begin{cases} 2^h + n2^{h+1} & \mu h \forall n \exists s R(x, h, n, s) \\ \uparrow & \text{otherwise} \end{cases}$ By the parametrization lemma[4] there is a total recursive g such that $\phi_{g(x)}(n) = f(x, n)$.
Then $\bigcup_{h \in \mathbb{N}} \{2^h + n2^{h-1} : n \in \mathbb{N}\} \cap W_{g(x)} \neq \emptyset$.

$x \in B \Rightarrow \exists h \forall n f(x, n) = 2^h + 2^{h+1}$
$\Rightarrow \exists h \forall \phi_{g(x)}(n) = 2^h + n2^{h+1}$
$\Rightarrow \exists h \forall n (2^h + n2^{h+1} \in W_g(x))$
$\Rightarrow g(x) \in A$

Thus $x \in B$ iff $g(x) \in A$. Therefore every Σ_3^0 subset of $\mathbb{N}$ is many-one reducible to A.

4 The Polynomial hierarchy [2][3]

4.a Definitions

The polynomial hierarchy generalizes the definitions of $NP, coNP, \Sigma_2^p, \Pi_2^p$ to all the languages that can be defined via a combination of a polynomial-time computable predicate and a constant number of $\forall/\exists$ quantifiers:

4.b Properties

4.c Examples

References

[1] M. Sipser. Introduction to the Theory of Computation. 1996.

[2] Arora and Barak. Computational complexity - a modern approach, 2009.

[3] D. Kozen. Theory of Computation. 2006.

[4] D. Marker. Lecture notes for math 502. 2015.