Theory of Computation: Polynomial Hierarchy

Problems not captured by NP

- Exact IndSet: Determine if the largest independent set of input graph G has size exactly k.
- No short certificate: How do you determine that all other independent set sizes are at most k?

Class Σ_2^p

- Set of all languages L for which there exists a polynomial time TM M and a polynomial q such that:
- $x \in L \iff \exists u \in \{0,1\}^{q(|x|)} \forall v \in \{0,1\}^{q(|x|)} M(x,u,v) = 1$ for all $x \in \{0,1\}^*$.
- Contains NP: M will ignore v no matter what it is.
- Contains coNP: M will take u to be the empty string.

Exact IndSet

- In Σ_2^p: There exists a size-k vertex subset S of G such that for all S' of size $k+1$, S is an independent set and S' is not an independent set.

Polynomial Hierarchy

- For each $i \geq 1$, a language L is in Σ_i^p if there exists a polynomial time deterministic TM M and a polynomial q such that:
- $x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} \ldots Q_i u_i \in \{0,1\}^{q(|x|)} M(x, u_1, \ldots, u_i) = 1$
  where Q_i denotes $\exists$ or $\forall$ depending on whether i is even or odd.
- Polynomial hierarchy PH $= \bigcup_i \Sigma_i^p$.

Polynomial Hierarchy

- $\Sigma_1^p = NP$
- For each i, define $\Pi_i^p = co\Sigma_i^p = \{\overline{L} | L \in \Sigma_i^p\}$.
  Thus, $\Pi_1^p = coNP$.
- For each i, $\Sigma_i^p \subseteq \Pi_{i+1}^p \subseteq \Sigma_{i+2}^p$.
  So, $PH = \bigcup_i \Pi_i^p$.

Hierarchy Collapse

- If $P = NP$ then $PH = P$.
- If $\Sigma_i^p = \Pi_i^p$ then $PH = \Sigma_i^p$: Similar proof.

If $P = NP$ then $PH = P$

- Proof by induction on i that $\Sigma_i^p, \Pi_i^p \subseteq P$.
  True for $i = 1$. Assume true for $i - 1$ and prove
  $\Sigma_i^p \subseteq P \implies \Pi_i^p \subseteq P$.
- Let $L \in \Sigma_i^p$. There is a TM M and polynomial q such that
  $x \in L \iff$
  $\exists u_1 \in \{0,1\}^{q(|x|)} \forall u_2 \in \{0,1\}^{q(|x|)} \ldots Q_i u_i \in \{0,1\}^{q(|x|)}$
  $M(x, u_1, u_2, \ldots, u_i) = 1$.
- Language L': $< x, u_1 > \in L' \iff$
  $\forall u_2 \in \{0,1\}^{q(|x|)} \ldots Q_i u_i \in \{0,1\}^{q(|x|)}$
  $M(x, u_1, u_2, \ldots, u_i) = 1$.
- So $L' \in \Pi_{i-1}^p \subseteq P$. So there is a polynomial time TM M' computing L'.
- Thus, $x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} M'(x, u_1) = 1$.
- So $L \in NP = P$.

Complete problems for PH

- A language L in Σ_i^p is Σ_i^p-complete if every $L' \in \Sigma_i^p$ reduces to L in polynomial time.
- Similarly PH-completeness.

Complete problems for PH

- PH-complete problem $\implies$ there is an i such that $PH = \Sigma_i^p$.
- Let L be PH-complete. L belongs to some Σ_i^p.

Complete problems on different levels

$\Sigma_i^p SAT = \exists u_1 \forall u_2 \ldots Q_i u_i \phi(u_1, u_2, \ldots, u_i) = 1$
where ϕ is an unquantified Boolean formula (may not be CNF).
This problem is Σ_i^p-complete.
Similarly, $\Pi_i^p SAT$ can be defined and is a Π_i^p-complete problem.

Alternating TM

- Like NDTM
- At each step there are two transitions to choose from
- Each state other than accept state t and halt state r is labelled with either $\exists$ or $\forall$.
- Suppose we are in a state labelled with $\exists$ then during this time, we have to find a sequence of choices that will carry the computation towards accept state t.
- Suppose we are in a state labelled with $\forall$ then during this time, we have to ensure that for all sequence of choices the machine is moving towards the accept state t.

Alternating Time

- For every $T : \mathbb{N} \to \mathbb{N}$, an alternating TM M runs in $T(n)$ time if for each input $x \in \{0,1\}^*$ and all possible sequence of transition choices M halts after at most $T(|x|)$ steps.
- $L = L(M)$ belongs to $ATIME(T(n))$: M is a $c.T(n)$-time ATM.

Acceptance in ATM

- Look at the configuration graph $G_{M,x}$.
- Labelled of some vertices of the graph as "accept": Any configuration where the state is t is labelled "accept".
  If a configuration C has the state labelled $\exists$ and there is an edge from C to C' labelled "accept", then label C as "accept".
  If a configuration C has the state labelled $\forall$ and C has edges to C_1 and C_2 both labelled "accept", then label C as "accept".
- M accepts x if the start configuration C_s is labelled "accept".

Fixed number of alternations

- $\Sigma_i TIME(T(n))$: Set of languages accepted by $c.T(n)$-time ATMs M with
  initial state s labelled $\exists$,
  and on any input x there are at most $i-1$ alternations from states with one label to states with another label on any directed path in $G_{M,x}$ starting from C_s.
- Similarly, $\Pi_i TIME(T(n))$.

PH and ATMs

For every $i \in \mathbb{N}$, $\Sigma_i^p = \bigcup_c \Sigma_i TIME(n^c)$ and $\Pi_i^p = \bigcup_c \Pi_i TIME(n^c)$.

Time-Space tradeoff for SAT

$\mathrm{TISP}(T(n), S(n))$ = set of languages decided by a TM M that on every input x takes at most $O(T(|x|))$ steps and uses at most $O(S(|x|))$ cells of its worktapes.

Time-Space tradeoff for SAT

Theorem: SAT $\notin$ TISP$(n^{1.1}, n^{0.1})$.

- SAT could still be in P
- SAT could still be in L or NL.
- This says that efficiency in both time and space resources is not possible for SAT.

Time-Space tradeoff for SAT

- Enough to show $\mathit{NTIME}(n) \not\subseteq \mathit{TISP}(n^{1.2}, n^{0.2})$:
- By Cook-Levin Theorem, any language in $\mathit{NTIME}(n)$ reduces to SAT in $n \mathrm{polylog} n$ time.
- If SAT $\in$ TISP$(n^{1.1}, n^{0.1})$, then by Cook-Levin Theorem we have that $\mathit{NTIME}(n) \subseteq \mathit{TISP}(n^{1.2}, n^{0.2})$.

Step 1: Relation to Alternations

- $\mathrm{TISP}(n^{12}, n^2) \subseteq \Sigma_2 TIME(n^8)$:
- Machine can use $c.n^{12}$ time and $c.n^2$ space, for some c.
- So each configuration described by $O(n^2)$ length string.
- Path from C_s to accepting configuration can be of length at most n^{12}.
- Necessary and sufficient:
  there exist n^6 configurations $C_0 = C_s, \ldots C_{n^6}$(accepting config.) such that
  for every $i \in [n^6]$ C_i can be computed from C_{i-1} within n^6 steps. [This can be verified in $O(n^7)$ time by checking reachability from C_{i-1}].
- 2 alternations in quantifiers- $O(n^8)$-time Σ_2 TM for deciding membership in L (length of the certificate for the n^6 C_i's is $O(n^8)$).

Step 2: Replacing Alternations with Time

- For contradiction, suppose $NTIME(n) \subseteq TISP(n^{1.2}, n^{0.2})$ $(\subseteq DTIME(n^{1.2}))$.
- Then we show that $\Sigma_2 TIME(n^8) \subseteq NTIME(n^{9.6})$:
- $L \in \Sigma_2 TIME(n^8) \iff$ there is a deterministic TM M such that $x \in L \iff$
  $\exists u \in \{0,1\}^{c|x|^8} \forall v \in \{0,1\}^{d|x|^8} M(x,u,v) = 1$
  and M runs in $O(|x^8|)$ time.

Step 2: Replacing Alternations with Time

- Assumption: $NTIME(n) \subseteq DTIME(n^{1.2})$.
- $\exists v \in \{0,1\}^{d|x|^8} M(x,u,v) = 0$ is a language L' taking input $< x, u >$ and belonging in $NTIME(n^8)$.
- By assumption, L' also belongs in $DTIME((n^8)^{1.2}) = DTIME(n^{9.6})$ (by padding). Assume D is a deterministic algorithm that answers 1 when input $< x, u > \in L'$.
- This means that is $< x, u > \in \overline{L'}$, $D(x,u) = 0$. This happens when $\forall v \in \{0,1\}^{d|x|^8} M(x,u,v) = 1$.
- Thus, $x \in L \iff \exists u \in \{0,1\}^{c|x|} D(x,u) = 0$
  Thus, $L \in NTIME(n^{9.6})$.

Conclusion

- (Assumption) $NTIME(n) \subseteq TISP(n^{1.2}, n^{0.2})$
  $\implies NTIME(n^{10}) \subseteq TISP(n^{12}, n^2)$ (by padding)
- $\subseteq \Sigma_2 TIME(n^8)$ (Step 1)
- $\subseteq NTIME(n^{9.6})$ (Step 2)
- So $NTIME(n^{10}) \subseteq NTIME(n^{9.6})$ ($\rightarrow\leftarrow$ Nondeterministic Time Hierarchy Theorem).