Theory of Computation: Ladner's Theorem

Ladner's Theorem

- Assuming $P \neq NP$, there exists a language $L \in NP$ that is neither in P nor NP-complete.
- For a function $H : \mathbb{N} \to \mathbb{N}$,
  $SAT_H = \{\phi 0 1^{n^{H(n)}} | \phi \in SAT, |\phi| = n\}$.
- Will be looking at a specific function H.

Observations

1. $H(n)$ is a constant function: Then SAT_H is SAT with polynomial padding; So SAT_H *is NP-complete* - cannot be in P if $P \neq NP$.
   Note: Any constant function will make SAT_H NP-complete.
2. $H(n)$ tends to infinity as n tends to infinity: SAT_H cannot be NP-complete:
   Then there is some $O(n^i)$-time reduction from SAT,
   An n-length instance ϕ reduces to an $O(n^i)$ length instance of SAT_H of the form $\psi 01^{|\psi|^{H(|\psi|)}}$.
   Equivalent instance ψ of SAT must be $o(n)$ in length. Apply this reduction enough times to obtain a constant length equivalent instance of SAT $\implies$ SAT is in P $(\rightarrow\leftarrow)$.
   Note: Any growing function will make input lengths of SAT_H long enough to check in polynomial time if the CNF-SAT part is satisfiable.

Observations contd.

1. Will choose an H such that if $SAT_H \in P$ then $H(n) = O(1)$, otherwise $H(n)$ tends to infinity with n. Such a $SAT_H \in NP$ will be neither in P nor NP-complete.
2. Suppose $SAT_H \in P$. Then $H(n) \leq c$ for all n. By Observation 1, this implies $P = NP$.
3. Otherwise, suppose SAT_H is NP-complete. By Observation 2, this implies $SAT \in P$ and that $P = NP$.

The function

- $H : \mathbb{N} \to \mathbb{N}$: $H(n)$ is the smallest $i < \log \log n$ s.t $\forall x \in \{0, 1\}^*, |x| \leq \log n$, DTM M_i outputs $SAT_H(x)$ within $i|x|^i$ steps. If no such i exists then $H(n) = \log \log n$.
- Well-defined: Definition only relies on strings of length $\log n$. $H(n)$ can be computed in polynomial time.
- To prove: if $SAT_H \in P$ then $H(n) = O(1)$, otherwise $H(n)$ tends to infinity with n.

The function contd.

- $H(n)$ is the smallest $i < \log\log n$ s.t $\forall x \in \{0,1\}^*, |x| \leq \log n$, DTM M_i outputs $SAT_H(x)$ within $i|x|^i$ steps. If no such i exists then $H(n) = \log\log n$.
  To prove: if $SAT_H \in P$ then $H(n) = O(1)$, otherwise $H(n)$ tends to infinity with n.
- $SAT_H \in P \implies H(n) = O(1)$: Let M be a machine solving SAT_H in cn^c steps.
- M has infinite representations; there is a number $j \geq c$ s.t. $M = M_j$.
- For $n > 2^{2^j}$, by definition $H(n) \leq j$. So $H(n) = O(1)$.

The function contd.

- $H(n)$ is the smallest $i < \log\log n$ s.t $\forall x \in \{0,1\}^*, |x| \leq \log n$, DTM M_i outputs $SAT_H(x)$ within $i|x|^i$ steps. If no such i exists then $H(n) = \log\log n$.
  To prove: if $SAT_H \in P$ then $H(n) = O(1)$, otherwise $H(n)$ tends to infinity with n.
- Suppose $H(n)$ does not tend to infinity with n - There is some constant c s.t $H(n) \leq c$ for infinitely many n's
  $\implies SAT_H \in P$:
- There is an i s.t $H(n) = i$ for infinitely many n's.
- Consider TM M_i. M_i must solve SAT_H in in^i time. Otherwise, if there is an input x where M_i gives the wrong answer, then $\forall n > 2^{|x|}$, $H(n) \neq i$ $(\rightarrow\leftarrow)$.