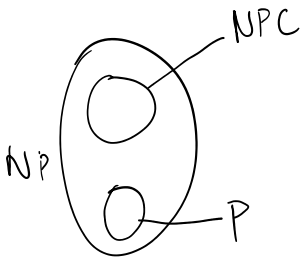
# Cook Levin Theorem

11 April 2021 23:28

<u>Theorem</u>: SAT is NP-Complete

We use a "computation histories" - kind of idea here.




We have to show

(1) $SAT \in NP$

(2) $\forall A \in NP, \quad A \leq_p SAT$.

<u>$SAT \in NP$</u>: Guess TRUE / FALSE for $x_1, x_2 \ldots x_n$.

Verify if $\phi$ is satisfied for the guessed assignment.

The main part is (2). We must show that any language $A \in NP$ must reduce to SAT.

That is, we need $f$ such that

$$w \in A \iff f(w) \in SAT.$$

All that we know about $A$ is that $A \in NP$.

We cannot assume any other structure of a any specific language / problem.

<u>$A \in NP$</u>: $A$ is decided by an NTM $N$ in time $n^k$.

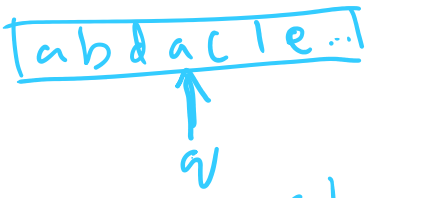
When $w \in A$, there is a sequence of computations that leads $N$ to accept $w$. This sequence has $\leq n^k$ steps. When $w \notin A$, no sequence of $(\ldots, a, b, R), (s, b, L), \ldots$

$\leq n^k$ steps. When $w \notin A$, no sequence of computations accept.

$\delta(q, a) = \{(q, b, R), (s, b, L), (s', c, R), \dots\}$

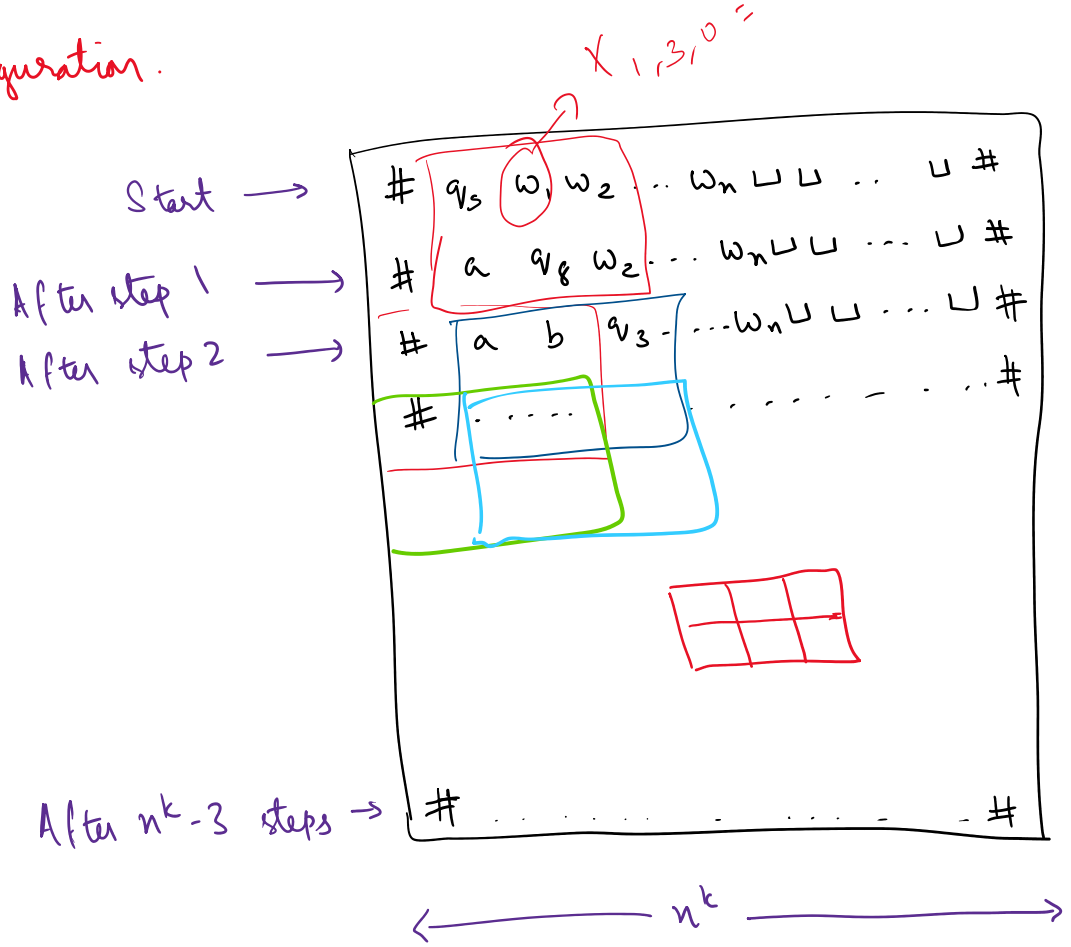
Let $N = (Q, \Sigma, \Gamma, \delta, q_s, q_a, q_r)$.

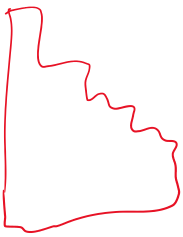
Let $N$ be a 1-tape NTM, runs in time $\leq n^k$ (actually $n^k - 3$). Let us define $\Delta = Q \cup \Gamma \cup \{\#\}$.




Gadget for reduction: Table or Tableau

Successive configurations. Each row is a configuration.



If the TM $N$ accepts/rejects before $n^k$ steps, the configuration remains unchanged after that.

If the TM [illegible]

the configuration remains unchanged after that.

We will create a formula $\phi$ which uses Boolean logic to check whether the tableau represents an accepting computation for N on w.

If there exists a path for N to accept w, then $\phi$ will have a satisfying assignment. Else $\phi$ won't have a sat. assignment.

$\phi$ checks the following:

(1) Does N start correctly?

(2) Does N move correctly?

(3) Does N end correctly?

And (4) we also need to check if the variables form a "proper encoding" of the table.

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}.$$

We have an $n^k \times n^k$ table. We have $|\Delta|$ variables that can occupy each cell.

$$x_{i,j,\ell} = \begin{cases} \text{TRUE} & \text{if } (i,j)^{th} \text{ cell has entry } \ell \\ \text{FALSE} & \text{if } (i,j)^{th} \text{ cell has entry} \neq \ell \end{cases}$$

$1 \le i, j \le n^k$ $\quad$ $\ell \in \Delta$

$\phi_{cell}$ : Is the table properly encoded.

$$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} \left[ \underbrace{\left( \bigvee_{\ell \in \Delta} x_{i,j,\ell} \right)}_{\text{Cell } (i,j) \text{ contains at least one entry}} \wedge \underbrace{\left( \bigwedge_{\substack{\ell, \ell' \in \Delta \\ \ell \neq \ell'}} \left[ \overline{x_{i,j,\ell}} \vee \overline{x_{i,j,\ell'}} \right] \right)}_{\text{Cell } (i,j) \text{ does not contain } > 1 \text{ entry.}} \right]$$

So $\phi_{cell}$ ensures that each cell $(i,j)$ contains exactly one member of $\Delta$.

---

$\phi_{start}$ : First row is a legal starting config for $N$ on $w$.

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_{start}} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge$$

$$\ldots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \ldots$$

$$\ldots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$$

$\phi_{accept}$ : The table represents an accepting computation.

$$\phi_{accept} = \bigvee_{1 \le i, j \le n^k} \left( x_{i,j,q_{accept}} \right)$$

$\phi_{move}$ : This is the hardest. We have to check that each configuration legally follows from the previous one.

The main idea here is that it is enough to check all the 2X3 windows. We say a window is valid if it is part of a valid transition.

$$\frac{b \quad q \quad a}{b \quad d \quad r}$$ when $(r, d, R) \in \delta(q, a)$ $\forall b \in \Gamma \cup \{\#\}$

$$\frac{b \quad q \quad a}{r \quad b \quad d}$$ when $(r, d, L) \in \delta(q, a)$ $\forall b \in \Gamma$

$$\frac{\# \quad q \quad a}{\# \quad \quad d}$$ when $(r, d, L) \in \delta(q, a)$

$$\begin{array}{ccc} \# & q & a \\ \hline \# & r & d \end{array}$$ when $(r, d, L) \in \delta(q, a)$

---

$$\begin{array}{ccc} b & c & q \\ \hline b & c & d \end{array}$$ if $(r, d, R) \in \delta(q, a)$ for some $a \in \Gamma$. where $b \in \Gamma \cup \{\#\}$, $c \in \Gamma$.

---

$$\begin{array}{ccc} q & a & b \\ \hline d & r & b \end{array}$$ if $(r, d, R) \in \delta(q, a)$ where $b \in \Gamma \cup \{\#\}$

---

$$\begin{array}{ccc} a & b & c \\ \hline a & b & c \end{array}$$ where $a \in \Gamma \cup \{\#\}$, $b, c \in \Gamma$ OR $a, b \in \Gamma$, $c \in \Gamma \cup \{\#\}$

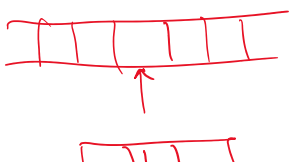
$$\begin{array}{ccc} a & b & c \\ \hline a & b & r \end{array}$$

$$\begin{array}{ccc} q & a & b \\ \hline a' & b & c \end{array}$$

---

Like this, one can list all the valid windows. No. of valid windows is finite, and depends only on N. It is independent of $|w|$.

<u>Claim</u>: If all $2 \times 3$ windows are valid, then all $C_i$ are legally followed by $C_{i+1}$.



Proof: If a symbol is not adjacent to

Proof: 1) If a symbol is not adjacent to the state, then it is unchanged.

| | | |
|---|---|---|
| a | b | c |
| | b | |

Middle symbol is unchanged in windows which don't have state.

| | | |
|---|---|---|
| a | q | b |
| | | |

2) If symbol is adj. to state, it appears in a window where state is in the middle.

This window captures a valid move for N.

---

$$\phi_{move} = \bigwedge_{1 \le i, j \le n^k} (i,j)^{th} \text{ window is legal.}$$

$$\bigvee \left( x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6} \right)$$

| | | |
|---|---|---|
| $a_1$ | $a_2$ | $a_3$ |
| $a_4$ | $a_5$ | $a_6$ |

is a valid window

$\phi \in CAT \iff N$ accepts $w$

is a valid window

We have shown $\phi \in SAT \iff N$ accepts $w$

$\iff w \in A$.

We also need : $\phi$ is constructible in poly time

There are $n^k \times n^k \times |\Delta|$ variables $= O(n^{2k})$

$\hookrightarrow$ ind. of $|w|$.

$\phi_{cell} = n^{2k}$ length

$\phi_{start} = n^k$ length

$\phi_{accept} = n^{2k}$ length

$\phi_{move} = \left.\begin{array}{c}\text{no. of valid} \\ \text{windows}\end{array}\right\} \times n^{2k} \times 6$

$= O(n^{2k})$

So $\phi$ is poly size in $n$. There is an easy algorithm to generate $\phi$.

---

## 3- SAT is NP- complete.

One way to show : Show SAT $\leq_p$ 3-SAT.

Slightly easier : To modify above proof so as to generate a 3-CNF SAT instance.

First show that the $\phi$ generated can be modified into a CNF instance.

<u>Q1</u>: Show that each of $\phi_{cell}$, $\phi_{start}$, $\phi_{accept}$, $\phi_{move}$ can be written in CNF form.

<u>Q2</u>: Show that any CNF formula has an equivalent 3-CNF expression.