Cook's theorem

Language L is

- *NP-hard* if *every* language in NP is polynomial-time reducible to L.

- *NP-complete* if $L \in$ NP and L is NP-hard.

COROLLARY Let L be any NP-hard language. L cannot be in P unless P $=$ NP.

THEOREM SAT is NP-complete.

Know: SAT $\in$ NP, so have to show SAT is NP-hard, i.e.,

- show how to reduce to SAT (in polynomial time) *every* $L \in$ NP!

Given: $L \in$ NP so there is polynomial time NTM M that accepts L.

Reduction takes M and input x for M and produces CNF formula $\phi = \phi_M(x)$ s.t.

$$M \text{ accepts } x \iff \phi_M(x) \text{ is satisfiable.}$$

ϕ has length $O(p(|x|)^2)$ where p bounds runtime for M.

Useful idea: View each computation sequence of M as a tableau.

Example: Consider simple NTM that recognizes all binary strings with at least two consecutive zeroes.

$$(q_0, 0, q_0, 0, R), (q_0, 1, q_0, 1, R)$$
$$(q_0, 0, q_1, 0, R), (q_1, 0, q_f, 0, R)$$

Runtime $p(n) = n$.

	0	1	2	3	4	5	6	7
0	#	q_0	0	1	0	0	$\overline{b}$	#
1	#							#
2	#							#
3	#							#
4	#							#

Tableau has $p(n)+1$ rows and $p(n)+4$ columns.

$(q_0, 0, q_0, 0, R), (q_0, 1, q_0, 1, R)$

$(q_0, 0, q_1, 0, R), (q_1, 0, q_f, 0, R)$

	0	1	2	3	4	5	6	7
0	#	q_0	0	1	0	0	$\not b$	#
1	#	0	q_1	1	0	0	$\not b$	#
2	#							#
3	#							#
4	#							#

	0	1	2	3	4	5	6	7
0	#	q_0	0	1	0	0	$\not b$	#
1	#	0	q_1	1	0	0	$\not b$	#
2	#	0	q_1	1	0	0	$\not b$	#
3	#	0	q_1	1	0	0	$\not b$	#
4	#	0	q_1	1	0	0	$\not b$	#

$$(q_0, 0, q_0, 0, R), (q_0, 1, q_0, 1, R)$$
$$(q_0, 0, q_1, 0, R), (q_1, 0, q_f, 0, R)$$

	0	1	2	3	4	5	6	7
0	#	q_0	0	1	0	0	$\overline{b}$	#
1	#	0	q_0	1	0	0	$\overline{b}$	#
2	#	0	1	q_0	0	0	$\overline{b}$	#
3	#	0	1	0	q_1	0	$\overline{b}$	#
4	#	0	1	0	0	q_f	$\overline{b}$	#

Generally:

	0	1	2	...	$n+1$	$n+2$	...		...	$m+2$	$m+3$
0	#	q_I	x_0	...	x_{n-1}	$\not b$	...		...	$\not b$	#
1	#										#
2	⋮										⋮
	⋮										⋮
m	#										#

where $m = p(n)$.

$\phi_M(x)$ expresses existence of an accepting tableau.

Boolean variables: Z_{ijs} where

- $0 \leq i \leq m$,
- $0 \leq j \leq m + 3$,
- s ranges over the set $\Gamma \cup Q \cup \{\#\}$.

Intention: Z_{ijs} true if and only if square (i, j) contains symbol s.

Construction of ϕ: conjunction of four subformulas

$$\phi = \phi_{\text{config}} \wedge \phi_{\text{initial}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{compute}}.$$

(i) ϕ_{config} ensures that each square contains precisely one symbol, i.e.,

$$\phi_{\text{config}} = \bigwedge_{i,j} [(\bigvee_{s} Z_{ijs}) \wedge \bigwedge_{s \neq s'} (\neg Z_{ijs} \vee \neg Z_{ijs'})].$$

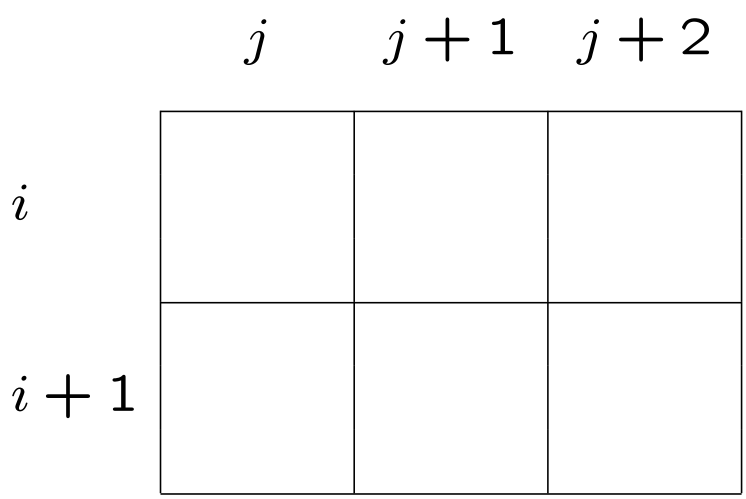
(ii) ϕ_{initial} ensures that first row is initial configuration of M on x, i.e.,

$$\phi_{\text{initial}} = Z_{00\#} \wedge Z_{01q_I} \wedge Z_{0,m+3,\#} \wedge \bigwedge_{j=2}^{n+1} Z_{0jx_{j-2}} \wedge \bigwedge_{j=n+2}^{m+2} Z_{0j\bar{b}}.$$

(iii) ϕ_{accept} ensures that final configuration is accepting, i.e.,

$$\phi_{\text{accept}} = \bigvee_{j} Z_{mjq_F}.$$

(iv) ϕ_{compute} ensures correctness of computations by stating that every 2×3 window is correct:



$$\phi_{\text{compute}} = \bigwedge_{\substack{0 \leq i \leq m-1 \\ 0 \leq j \leq m+1}} C_{ij}$$

Generally: C_{ij} based on a fixed formula that enumerates all legitimate windows allowed by transition relation δ of M; various C_{ij} differ only in values of coordinate parameters i, j.

Example:

$$(q_0, 0, q_0, 0, R), (q_0, 1, q_0, 1, R)$$
$$(q_0, 0, q_1, 0, R), (q_1, 0, q_f, 0, R)$$

Allowed:

q_0	0	0
0	q_0	0

q_0	0	0
0	q_1	0

q_0	0	1
0	q_0	1

q_0	0	1
0	q_1	1

#	q_0	0
#	0	q_0

#	q_0	0
#	0	q_1

q_0	0	#
0	q_0	#

q_0	0	#
0	q_1	#

Not allowed:

q_1	1	0
1	q_1	0

q_0	$\bar{b}$	0
0	q_1	0

q_0	$\bar{b}$	1
$\bar{b}$	q_0	1

q_0	0	1
0	q_1	$\bar{b}$

q_0	#	0
#	0	q_0

#	q_0	0
0	#	q_1

#	q_0	0
0	q_0	#

q_0	0	0
q_1	#	#

Finally: Everything in CNF except for the C_{ij}.

• Each C_{ij} *independent* of input x to M.

• Can put into CNF by brute force; cost is constant (remember M is fixed).

Conclusion: ϕ has length $O(p(n)^2)$ and can be built in polynomial time.

THEOREM Let L_1 and L_2 be languages. If L_1 is NP-hard and $L_1 \leq_{\mathsf{P}} L_2$, then L_2 is NP-hard.

THEOREM CLIQUE is NP-complete.