Cook-Levin Theorem

Theorem (Cook 1971, Levin) SAT (in CNF form) is NP-complete

We have already shown that SATISFIABILITY is in NP. So now we need to show that for *any* language $L \in NP$, L can be reduced to SATISFIABILITY

So for, we have only shown reducibilities of a *single* problem to another. How can we handle all problems in NP?

Give a *generic* reduction, based on nondeterministic TM's:

- For any $L \in NP$, there must be a polynomial time nondeterministic TM M which accepts L.
- We will use this fact to show that for any $L \in NP$, there is a polynomial time reduction f_L such that for any x, $x \in L$ if and only if $f_L(x)$ is satisfiable.

Defining f_L

Suppose $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept})$, and that p is a polynomial which bounds the running time of M. Assume that $p(n) \geq n$.

Suppose that Q is numbered as follows: $q_0, q_1, \ldots, q_w$, where $q_1 = q_{accept}$. and that Γ is numbered $s_0, s_1, \ldots s_v$, where $s_0 = \sqcup$.

We will number the tape cells $\ldots, -2, -1, 0, 1, 2, \ldots$. Note that if the running time of M is bounded by $p(n)$ then we can never move right or left from cell 0 more than $p(n)$ times, and so we never need to consider tape squares with a number whose absolute value is higher than $p(n)$.

We now show how to create $f_L(x)$ for any instance x of L. Let $n = |x|$.

The Variables

We first specify the set of variables.

Variable	Range	Intended meaning
$y_{i,k}$	$1 \leq i \leq p(n)$ $0 \leq k \leq w$	At time i, M is in state q_k
$h_{i,j}$	$1 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)$	At time i, tape head is at cell j
$r_{i,j,k}$	$1 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)$ $0 \leq k \leq v$	At time i, tape cell j contains symbol s_k

The Clause Groups

The clauses come in six groups, each of which impose a constraint on satisfying truth assignments which force a legal accepting computation.

The Clause Groups

Clause group	Restriction
G_1	At each time i, M is in exactly one state
G_2	At each time i, the tape head is on exactly one cell
G_3	At each time i, each tape cell contains exactly one symbol
G_4	At time 1, the computation is in its initial configuration
G_5	By time $p(n)$, M has entered state q_1
G_6	For each time i, every configuration at time $i+1$ follows in one step from the configuration at time i, according to δ

Inside the Clause Groups

We will now take a look inside some of the clause groups. For G_1 we need for every $i, \quad 1 \leq i \leq p(n)$ a clause

$$\{y_{i,0}, y_{i,1}, \dots, y_{i,w}\}$$

which says that we are in *some state*, and also for every pair j, j', $1 \leq j < j' \leq w$ we need a clause

$$\{\overline{y_{i,j}}, \overline{y_{i,j'}}\}$$

which says that we are not in *both* states q_j and $q_{j'}$.

Groups G_2 and G_3 are similar. Group G_5 just contains the single clause $\{y_{p(n),1}\}$, which says that M is in the accepting state q_1 at time $p(n)$.

Inside Group 4

G_4 is made up of the following clauses:

$\{y_{1,0}\}$ - M starts in state q_0

$\{h_{1,0}\}$ - M starts scanning cell 0

$\{r_{1,-p(n),0}\}, \{r_{1,-p(n)+1,0}\}, \ldots, \{r_{1,-1,0}\},$
$\{r_{1,0,k_1}\}, \ldots, \{r_{1,n-1,k_n}\},$
$\{r_{1,n,0}\}, \ldots, \{r_{1,p(n)+1,0}\}$

-The initial tape is $s_{k_1} \ldots s_{k_n}$ followed by $\sqcup$'s, where $x = s_{k_1} s_{k_2} \ldots s_{k_n}$

(NOTE: this last clause is the only one which depends on the actual value of x (compare to the Sudoku encoding!))

Inside Group 6

This is the most complicated. Basically we need to say that every configuration at step $i+1$ must follow in one step from a configuration at step i.

First note the following fact about propositional logic: in general, an implication of the form

$(z_1 \wedge z_2 \wedge \cdots \wedge z_k) \rightarrow y$

is equivalent to the clause $\{\overline{z_1}, \ldots, \overline{z_k}, y\}$

Inside Group 6

There are two subgroups here. The first just say that at any time i, if cell j is not being scanned, then it will be *unchanged* at time $i+1$. This is expressed by having the following clauses for all i, j, k where $1 \leq i < p(n)$, $-p(n) \leq j \leq p(n)$, $0 \leq k \leq v$: $\{\overline{r_{i,j,k}}, h_{i,j}, r_{i+1,j,k}\}$

(In implicational form, this is $(r_{i,j,k} \wedge \overline{h_{i,j}}) \rightarrow r_{i+1,j,k}$)

Inside Group 6

The remaining subgroup in G_6 depends on the transition function δ, e.g., suppose that $\delta(q_m, s_k) = \{(q_{m'}, s_{k'}, R)\}$. Then we will have the following clauses for all i, $0 \leq i \leq p(n)$ and all j, $-p(n) \leq j \leq p(n)$:

$\{\overline{y_{i,m}}, \overline{h_{i,j}}, \overline{r_{i,j,k}}, y_{i+1,m'}\}$
$\{\overline{y_{i,m}}, \overline{h_{i,j}}, \overline{r_{i,j,k}}, h_{i+1,j+1}\}$
$\{\overline{y_{i,m}}, \overline{h_{i,j}}, \overline{r_{i,j,k}}, r_{i+1,j,k'}\}$

These again arise from implications, for example the first set from:

$(y_{i,m} \wedge h_{i,j} \wedge r_{i,j,k}) \rightarrow y_{i+1,m'}$

What happens if there's more than one choice for δ?

More than one choice for transition

Suppose for $\delta(q_m, s_k)$ there are T nondeterministic choices. For each possible value of i and j, add T variables $z_{i,j,k,m,1}, z_{i,j,k,m,2}, \ldots, z_{i,j,k,m,T}$

Now add a clause

$\{\overline{y_{i,m}}, \overline{h_{i,j}}, \overline{r_{i,j,k}}, z_{i,j,k,m,1}, z_{i,j,k,m,2}, \ldots, z_{i,j,k,m,T}\}$

which corresponds to the implication

$(y_{i,m} \wedge h_{i,j} \wedge r_{i,j,k}) \rightarrow (z_{i,j,k,m,1} \vee z_{i,j,k,m,2} \vee \cdots \vee z_{i,j,k,m,T})$

Finally, for each possible value of i, j, t add clauses of the form $\{\overline{z_{i,j,k,m,t}}, y_{i+1,m'}\}$, $\{\overline{z_{i,j,k,m,t}}, h_{i+1,j'}\}$, and $\{\overline{z_{i,j,k,m,t}}, r_{i+1,j,k'}\}$, where the exact values of m', j' and k' depend on the details of the tth alternative.

f_L is Polynomially Bounded

It is not hard to see that:

- The number of clauses in each group is either constant (depends only on M) or polynomial in $n = |x|$
- The size of any clause is polynomial in $n = |x|$ (note: all clauses except the clause in G_4 which specifies the input have a constant number of clauses. Each variable can be encoded with polynomial in n many bits.)

f_L is a reduction from L to SAT

- $w \in L$ iff $f_L(w) \in SAT$.

This is clear from the definition of the reduction.