Efficient Computations

P $= \{L$: some poly time bounded deterministic Turing machine accepts $L\}$.

NP $= \{L$: some poly time bounded nondeterministic Turing machine accepts $L\}$.

coNP$=\{L : \overline{L} \in$ **NP**$\}$.

P=**NP**?

Proposition: Suppose $L \in \mathbf{NP}$.
Then there exists a (deterministic) polynomial time computable predicate $P(x, y)$, and a polynomial $q(\cdot)$ such that
$x \in L$ iff $(\exists y : |y| \leq q(|x|))[P(x, y)]$.

Proof: Suppose N is a $q(n)$ time bounded NDTM accepting L. Without loss of generality assume that N has exactly two choices in each state.

$P(x, y)$ is defined as follows.

- Let $y = y_1 y_2 \cdots y_m$.
- If $m > q(|x|)$ then reject.
- Otherwise simulate N, where at step i, choose the next state based on whether y_i is 0 or 1.
- $P(x, y)$ is 1 iff N accepts in the above simulation.

Now, $(\exists y : |y| \leq q(|x|))[P(x, y)]$ iff $N(x)$ has an accepting path.

In the proposition one often calls y such that $P(x, y) = 1$ as a "certificate" or "proof" that $x \in L$.

Thus one can consider **NP** as class of languages for which "proofs" can be easily (in polynomial time) verified.

Reducibility

$L_1 \leq_m^p L_2$ (read: L_1 is poly time, many-one, reducible to L_2): there exists poly time computable function f such that $x \in L_1 \Leftrightarrow f(x) \in L_2$.

$L_1 \leq_T^p L_2$ (read: L_1 is poly time, Turing, reducible to L_2): there exists a polynomial time oracle Turing machine M, such that the M^{L_2} accepts L_1.

$L_1 \leq_m^{\text{log space}} L_2$ (read: L_1 is log-space many-one reducible to L_2): there exists a function f, which is computable by a log space bounded Turing machine, such that $x \in L_1 \Leftrightarrow f(x) \in L_2$.

NP-completeness

A set L is said to be **NP**-complete iff
(1) $L \in \mathbf{NP}$, and
(2) $(\forall L' \in \mathbf{NP})[L' \leq_m^p L]$.

If (2) is satisfied, then the problem is said to be **NP**-hard.
The interest in **NP**-complete problems arises from the fact that many of the interesting combinatorial problems are **NP**-complete.

Some famous NP complete problems.

1. Satisfiability:
INSTANCE: A set U of variables and a collection C of clauses over U.
QUESTION: Is there a satisfying truth assignment for C?

2. 3-Dimensional Matching:
INSTANCE: Three disjoint finite sets X, Y, Z, each of cardinality n, and a set $S \subseteq X \times Y \times Z$.
QUESTION: Does S contain a matching? i.e. is there a subset $S' \subseteq S$ such that $card(S') = n$ and no two elements of S' agree in any coordinate?

3. Vertex Cover:
INSTANCE: A graph $G = (V, E)$ and a positive integer $K \leq card(V)$.
QUESTION: Is there a vertex cover of size K or less for G? i.e. is there a subset $V' \subseteq V$ such that, $card(V') \leq K$ and for each edge $(u, v) \in E$, at least one of u, v belongs to V'?

4. MAX-CUT:
INSTANCE: An undirected graph $G = (V, E)$, and a positive integer $K \leq card(E)$.
QUESTION: Is there a cut of G with size $> K$? Here (X, Y) is said to be a cut of G, if (X, Y) is a partition of V. That is, $X \cap Y = \emptyset$ and $X \cup Y = V$. Size of a cut (X, Y) of G, is $card(\{(v, w) : v \in X$ and $w \in Y$ and $(v, w) \in E\})$. That is, size of a cut (X, Y) is the number of edges in G which connect X and Y.

5. Clique:
INSTANCE: A graph $G = (V, E)$ and a positive integer $K \leq card(V)$.
QUESTION: Does G contain a clique of size K or more? i.e. is there a subset $V' \subseteq V$, such that $card(V') \geq K$, and for all distinct $u, v \in V'$, $(u, v) \in E$?

6. Hamiltonian Circuit:
INSTANCE: A graph $G = (V, E)$
QUESTION: Does G contain a Hamiltonian circuit? i.e. is there a simple circuit which goes through all the vertices of G?

7. Partition:
INSTANCE: A finite set A and a size $s(a) > 0$, for each $a \in A$.
QUESTION: Is there a subset A' of A such that $\Sigma_{a \in A'} s(a) = \Sigma_{a \in A - A'} s(a)$?

8. Set Cover:
INSTANCE: A finite set A, a collection $\{S_1, S_2, \ldots, S_m\}$ of subsets of A, and a number k.
QUESTION: Is there a subset Y of $\{1, \ldots, m\}$, of size at most k, such that $A \subseteq \cup_{i \in Y} S_i$.

9. Traveling Salesman Problem:
INSTANCE: A complete weighted graph $G = (V, E)$, and a bound B.
QUESTION: Is there a Hamiltonian circuit of weight $\leq B$?

Satisfiability (SAT) is **NP**-complete

Theorem (Cook): Satisfiability is **NP**-complete.
Proof sketch:
SAT is in **NP**: guess a satisfying truth assigment TA, and then verify by checking that each of the clauses has at least one true literal.

To show: for any L in **NP**, $L \leq_m^p SAT$.
Suppose $L \in \mathbf{NP}$.
Let P be a polynomial time computable predicate such that
$x \in L$ iff $(\exists y : |y| \leq q(|x|))[P(x, y)]$.
Let M be $p(n)$ time bounded machine which decides P (i.e. M accepts on input x, y iff $P(x, y) = 1$).
Below we use n for $|x|$.
Wlog M uses two tapes, and initially the two inputs are on the 2 tapes (called input and guess tape).

Alphabet set of M: $\Sigma = \{a_0, \ldots, a_r\}$, where a_0 stands for "blank". States of M: $Q = \{q_0, \ldots, q_s\}$, where q_0 is starting state, q_1 is the accepting state and q_2 is rejecting state.
We assume that once M reaches the accepting or rejecting state it just loops in that state.

What we plan to do is mimic the computation of the machine from time $t = 0$ (start) to time $t = p(n)$.
The function f reducing L to satisfiability is as follows.
$f(x = x_1 x_2 \cdots x_n) = (U, G)$, where the set of variables U and the set of clauses G is described below. It can be easily verified that this reduction can be done by a polynomial time bounded Turing machine.

Set of Variables, U

For $0 \leq t \leq p(n)$, we have the following variables in U.

$Q[t, q_i]$, for $q_i \in Q$. Intuitively, $Q[t, q_i]$ being true will denote the fact that at time t, M is in state q_i.

$H_1[t, l]$, for $1 \leq l \leq p(n) + 1$. Intuitively, $H_1[t, l]$ being true will denote the fact that at time t, the head on first tape of M is at location l.

$H_2[t, l]$, for $1 \leq l \leq p(n) + 1$. Intuitively, $H_2[t, l]$ being true will denote the fact that at time t, the head on second tape of M is at location l.

$C_1[t, l, a]$, for $1 \leq l \leq p(n) + 1$, $a \in \Sigma$. Intuitively, $C_1[t, l, a]$ being true will denote the fact that at time t the contents of l-th cell in the first tape is a.

$C_2[t, l, a]$, for $1 \leq l \leq p(n) + 1$, $a \in \Sigma$. Intuitively, $C_2[t, l, a]$ being true will denote the fact that at time t the contents of l-th cell in the second tape is a.

Clauses

G consists of the following clauses divided in 6 groups for ease of presentation/understanding.

1. Clauses for "exactly one state at time t"

For $0 \leq t \leq p(n)$, we have a clause

$(Q[t, q_0] \vee Q[t, q_1] \vee \cdots \vee Q[t, q_s])$

(i.e. M is in at least one internal state at any time).

For $0 \leq t \leq p(n)$,

$(\neg Q[t, q_i] \vee \neg Q[t, q_j])$, for $0 \leq i < j \leq s$.

(i.e. M is not in two internal states at the same time).

Note that the above set of clauses ensure that M is in exactly one internal state at any time.

2. Clauses for "head at exactly one position at time t"
For $0 \le t \le p(n))$, For $1 \le i < j \le p(n)+1$, we have the clauses,

$$(H_1[t,1] \vee H_1[t,2] \vee \cdots \vee H_1[t,p(n)+1]),$$
$$(\neg H_1[t,i] \vee \neg H_1[t,j]),$$
$$(H_2[t,1] \vee H_2[t,2] \vee \cdots \vee H_2[t,p(n)+1]), \text{ and}$$
$$(\neg H_2[t,i] \vee \neg H_2[t,j]).$$

3. Clauses for "exactly one symbol at time t in a cell"
For $0 \le t \le p(n)$, for $1 \le l \le p(n)+1$, $0 \le i < j \le r$, we have the clauses,

$$(C_1[t,l,a_0] \vee C_1[t,l,a_1] \vee \cdots \vee C_1[t,l,a_r]),$$
$$(\neg C_1[t,l,a_i] \vee \neg C_1[t,l,a_j]),$$
$$(C_2[t,l,a_0] \vee C_2[t,l,a_1] \vee \cdots \vee C_2[t,l,a_r]),$$
$$(\neg C_2[t,l,a_i] \vee \neg C_2[t,l,a_j]),$$

4. Clauses for initial state

$(Q[0, q_0])$,
$(H_1[0, 1])$, $(H_2[0, 1])$,
$(C_1[0, 1, x_1])$, . . . $(C_1[0, n, x_n])$, $(C_1[0, n{+}1, a_0])$, . . ., $(C_1[0, p(n), a_0])$.
$(C_2[0, q(n) + 1, a_0])$, . . ., $(C_2[0, p(n), a_0])$.
$(C_2[0, l + 1, a_0] \vee \neg C_2[0, l, a_0])$, for $1 \leq l < q(n)$ (to disallow blanks in "y").

Note that we have not specified the value of y in the guess tape! This allows any arbitrary initial content of guess tape, with length at most $q(n)$.

5. Clause for final state
$(Q[p(n), q_1])$.

6. Clauses for orderly transition

First we need to make sure that symbols do not change at locations where the head is not there.

For $0 \leq t < p(n)$ and $1 \leq l \leq p(n)+1$, we have the clauses,

$$(H_1[t,l] \vee C_1[t,l,a] \vee \neg C_1[t+1,l,a]), \text{ for } a \in \Sigma.$$

$$(H_2[t,l] \vee C_2[t,l,a] \vee \neg C_2[t+1,l,a]), \text{ for } a \in \Sigma.$$

Now we give the clauses which ensure the transition based on the transition table of M.
Suppose $(q, a, b, q', a', b', m_1, m_2)$ is an entry in the transition table of M.
Then we have the following clauses.
For $0 \leq t < p(n)$, $1 \leq j, j' \leq p(n)$.

$$(\neg H_1[t, j] \vee \neg H_2[t, j'] \vee \neg Q[t, q] \vee \neg C_1[t, j, a] \vee \neg C_2[t, j', b] \vee Q[t+1, q'])$$
$$(\neg H_1[t, j] \vee \neg H_2[t, j'] \vee \neg Q[t, q] \vee \neg C_1[t, j, a] \vee \neg C_2[t, j', b] \vee H_1[t + 1, j + m_1])$$
$$(\neg H_1[t, j] \vee \neg H_2[t, j'] \vee \neg Q[t, q] \vee \neg C_1[t, j, a] \vee \neg C_2[t, j', b] \vee H_2[t + 1, j' + m_2])$$
$$(\neg H_1[t, j] \vee \neg H_2[t, j'] \vee \neg Q[t, q] \vee \neg C_1[t, j, a] \vee \neg C_2[t, j', b] \vee C_1[t+1, j, a'])$$
$$(\neg H_1[t, j] \vee \neg H_2[t, j'] \vee \neg Q[t, q] \vee \neg C_1[t, j, a] \vee \neg C_2[t, j', b] \vee C_2[t+1, j', b'])$$

We now show that the reduction works.

Note that the reduction can be computed in polynomial time.

Now suppose $f(x) = (U, G)$. We claim that $x \in L$ iff G is satisfiable.
Suppose $x \in L$. Then there exists a y such that $P(x, y)$ is true.
Thus M accepts on input (x, y).
Assign the truth values to variables based on the computation of M.
It is easy to verify that all the clauses must be satisfied.

Now suppose that (U, G) is satisfiable. Pick a satisfying assignment in above.
Let $C_2'[0, l] = a_i$ iff $C_2[0, l, a_i]$ is true in the above assignment.
Let $y = C_2'[0, 1]C_2'[0, 2] \cdots C_2'[0, q(n)]$, where we ignore the trailing blanks.
It is easy to verify that $M(x, y)$ must accept.
Thus $P(x, y)$ is true, and hence $x \in L$.

Proposition: $\leq_m^p$ is reflexive and transitive.
Proof:
Reflexive: Any L can be reduced to itself by identity function $f(x) = x$.

Transitive: Suppose $L_1 \leq_m^p L_2$ and $L_2 \leq_m^p L_3$.
Suppose f, g are polynomial time computable functions such that $x \in L_1 \Leftrightarrow f(x) \in L_2$ and $x \in L_2 \Leftrightarrow g(x) \in L_3$.
Let $h(x) = g(f(x))$. Clearly h is polynomial time computable.
Now $x \in L_1 \Leftrightarrow f(x) \in L_2 \Leftrightarrow g(f(x)) \in L_3$.
Thus $x \in L_1 \Leftrightarrow h(x) \in L_3$.
Thus $L_1 \leq_m^p L_3$. This shows that $\leq_m^p$ is transitive.

Corollary: If L is $\mathbf{NP}$-complete, $L' \in \mathbf{NP}$ and $L \leq_m^p L'$ then L' is $\mathbf{NP}$-complete.
The above corollary allows us to prove that a problem $L' \in \mathbf{NP}$ is $\mathbf{NP}$-complete by just showing that $L' \in \mathbf{NP}$ and some KNOWN $\mathbf{NP}$-complete problem is polynomial time, many one reducible to L'.