The Class NP

- Do T.S.P., Sub-Set Sum, MAX-cut, 3SAT, ... have a polytime algorithm?

- Not known.

- They do have a non-deterministic polytime algorithm.

Def A non-det TM $M$ is a tuple $M = (Q, \Sigma, \Gamma, \delta, q_{\text{start}}, q_{\text{accept}}, q_{\text{reject}})$

where $\delta : Q \times \Gamma \rightarrow (Q \times \Gamma \times \{L, R\})^2$.

I.e. An instruction is of form

$$(q, a) \overset{\text{or}}{\longrightarrow} (q', a', L) \quad \text{or} \quad (q'', a'', R).$$

Two (or more or none) choices at each step.

Note. We'll assume a time bound $t(n)$. 
Conventions (w.l.o.g.)
- Machine halts in time/#steps exactly \( t(n) \).
- Exactly two choices at each step.

Computation of a Non-det TM

\[ \begin{array}{c}
\text{q} \rightarrow \text{Rej} \rightarrow \text{Acc} \rightarrow \text{Rej} \rightarrow \text{Rej} \rightarrow \text{Acc} \rightarrow \text{Rej}
\end{array} \]

2 paths/leaves

Def. Input \( x \in \Sigma^* \) is accepted by a NTM M if on input \( x \), M has at least one computation that accepts.

Def. A language \( L \) is accepted by a NTM M in time \( t(n) \) if
- \( \forall x \in \Sigma^*, |x| = n \), M runs for time (at most) \( t(n) \).
- \( x \in L \Rightarrow M \) has at least one computation on \( x \) that accepts.
- \( x \notin L \Rightarrow \) Every computation of \( M \) on \( x \) rejects.

**Def**  \( \text{NTIME}(t(n)) := \) Class of languages accepted by NTMs in time \( O(t(n)) \).

**Note**  \( \text{NTIME}(t(n)) \subseteq \text{DTIME}(2^{t(n)}) \).

**Proof**  In det time \( 2^{t(n)} \), all \( 2^{t(n)} \) paths of a NTM can be simulated and then one checks if at least one path accepts. \( \blacksquare \)

**Def**  \( \text{NP} = \bigcup_{k=1,2,3,\ldots} \text{NTIME}(n^k) \).

**Clearly**  - \( \text{DTIME}(t(n)) \subseteq \text{NTIME}(t(n)) \).
- \( P \subseteq \text{NP} \).
Seemingly

\[ \text{NP} \]

- TSP
- 3SAT
- MAXCUT
- SUBSETSUM

\[ P \subset \text{NP} \]

- \( P \) vs \( \text{NP} \)
- Widely believed that \( P \neq \text{NP} \)

Problems in \( \text{NP} \)

All problems in \( P \)

\text{COMPOSITE} \in \text{NP}

\text{COMPOSITE} = \{ \langle n \rangle \mid n \text{ is composite} \}

Note that if \( k = \#\text{bits in } n \) then the input size is \( k \).

Following polytime NTM accepts \text{COMPOSITE}.

\( M \) := “on input \( \langle n \rangle \),
let \( k = \#\text{bits in } n \).
Non-deterministically choose a \( k \)-bit integer \( d \).
Accept if \( 2 \leq d < n \) and \( d \) divides \( n \).
Reject otherwise.”
Note that one can check in time $\text{poly}(k)$ whether $2 \leq d < n$ and whether $d$ divides $n$.

$\langle n \rangle \in \text{COMPOSITE} \Rightarrow \exists d, 2 \leq d < n, d \mid n$

$\Rightarrow \exists$ at least one accepting computation

$\langle n \rangle \notin \text{COMPOSITE} \Rightarrow \nexists d, 2 \leq d < n, d \mid n$

$\Rightarrow$ Every computation rejects.

$d = "\text{witness}"$ that $\langle n \rangle \in \text{COMPOSITE}$.

$d = "\text{proof}"$
\text{SUBSET-SUM} \in \text{NP}

\text{SUBSET-SUM} = \{ (a_1, a_2, \ldots, a_n ; t) \mid 
\begin{align*}
& a_1, a_2, \ldots, a_n, t \text{ are } n\text{-bit integers}, \\
& \exists S \subseteq \{a_1, \ldots, a_n\} \text{ s.t. } \sum_{i \in S} a_i = t.
\end{align*}
\}

Following poly\((n)\)-time NTM \(M\) accepts \text{SUBSET-SUM}.

\(M := "\text{On input } (a_1, \ldots, a_n ; t), \text{ Non-det choose a subset } S \subseteq \{a_1, \ldots, a_n\}. \text{ Accept if } \sum_{i \in S} a_i = t. \text{ Reject otherwise.}""

\[
S = \{ i \mid 1 \leq i \leq n, \text{ choice } '1' \text{ is made at step } i \}.
\]
Traveling Salesperson $\in$ NP

$TSP = \{ <G, l> \mid G$ is a weighted graph and has a tour of length $\leq l \}$

Following polytime NTM $M$ accepts $TSP$:

$M := $ "On input $<G, l>$,
Non-det choose a tour $\sigma$.
If weight of $\sigma$ is $\leq l$, accept.
Reject otherwise."

PRIMES $\in$ NP

$PRIMES = \{ <n> \mid n$ is a $k$-bit integer that is a prime. $\}$.

- How can I prove to you that $n$ is a prime (in time poly($k$))?
- Hmm… Highly non-trivial!

As it turns out PRIMES $\in$ P!"
Def: A (polytime) NTM M is said to be "guess & verify" machine if there is a polytime det machine M_{verifier} and

\[ M := \begin{array}{l}
\text{"On input } x, \ |x| = n, \\
\text{"Guess"} \{ \\
\text{Non-det select a string } y, \ |y| = n^{0(1)} \\
\text{Run } M_{verifier} \text{ on pair } (x, y) \\
\text{"verify"} \{ \\
\text{Accept iff } M_{verifier} \text{ accepts} \\
\end{array} \]

Note - All NTMs we saw are guess & verify m/c's.

- W.l.o.g. one can assume that NTMs are guess & verify m/c's (by first non-deterministically selecting string y denoting all choices and then simulating the path along those choices.

\[ y = \text{"witness"/"proof"} \]

- \[ m = \text{run time} \]
- \[ y_1y_2...y_m \in \{0,1\}^* \]
Reductions and NP-completeness

- We don't know if TSP, 3SAT, SUBSET-SUM, etc. have a (det) polytime algorithm.
- But we know that if one of them does, so do all others!
- So these are the “hardest problems” in NP!

**Reductions**

**Def** A TM $M$ is **det** with output has an extra, one-way output tape to write on. On input $x$, the contents of the output tape after the M/C halts is its output $M(x)$.

**Note** If $M$ is polytime then $|M(x)| = |x|^{O(1)}$. 
Def A language $A$ reduces in polynomial time to a language $B$ if there is a polytime dec TM $M$ with output s.t. 
$\forall x \in \Sigma^*$, on input $x$, $M$ outputs $y = M(x)$ and $x \in A \iff y \in B$.

This is denoted as $A \leq_p B$.

Claim If $A \leq_p B$ and $B \in P$ then $A \in P$.

Proof

Claim If $A \leq_p B$ and $B \in P$ then $A \in P$.

$$M = \text{Reduction}$$

Suppose $M^B$ decides $B$ in polytime.

Time $n^C \iff M^B_{\text{red}}$ is the reduction $A$ to $B$.
Then $M_A$ decides $A$ in time $n^{ck}$ as:

$M_A := \text{"On input } x, |x| = n, \text{ Run } M_{\text{red}} \text{ on } x \text{ to output } y, \text{ Using } M_B, \text{ decide if } y \in B. \text{ Accept iff } M_B \text{ accepts } y."$

**Correctness**

$x \in A \iff y \in B$.

**Runtime**

- $|y| \leq |x|^{ck} = n^{ck}$
- Runtime of $M_B$ on $y$ is $|y|^c \leq (n^{ck})^c = n^{ck}$.

Note similarly if $A \leq_p B, B \leq_p C$ then $A \leq_p C$.

**Proof**

\[ A \xrightarrow{\text{red}} B \xrightarrow{\text{red}} C \]

\[ x \mapsto y \mapsto z \]

$x \in A \iff y \in B \iff z \in C$. 