P vs NP

CS 154, Omer Reingold
Nondeterministic Turing Machines

...are just like standard TMs, except:

1. The machine may proceed according to several possible transitions (like an NFA)

2. The machine accepts an input string if there exists an accepting computation history for the machine on the string
Definition: A nondeterministic TM is a 7-tuple

\[ \Gamma = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}}) \], where:

- \( Q \) is a finite set of states
- \( \Sigma \) is the input alphabet, where \( \square \notin \Sigma \)
- \( \Gamma \) is the tape alphabet, where \( \square \in \Gamma \) and \( \Sigma \subseteq \Gamma \)
- \( \delta : Q \times \Gamma \rightarrow 2(\mathbb{Q} \times \Gamma \times \{L,R\}) \)
- \( q_0 \in Q \) is the start state
- \( q_{\text{accept}} \in Q \) is the accept state
- \( q_{\text{reject}} \in Q \) is the reject state, and \( q_{\text{reject}} \neq q_{\text{accept}} \)
Defining Acceptance for NTMs

Let $N$ be a nondeterministic Turing machine

An accepting computation history for $N$ on $w$ is a sequence of configurations $C_0, C_1, \ldots, C_t$ where

1. $C_0$ is the start configuration $q_0w$,
2. $C_t$ is an accepting configuration,
3. Each configuration $C_i$ yields $C_{i+1}$

Def. $N(w)$ accepts in time $t$ $\iff$ Such a history exists

$N$ has time complexity $T(n)$ if for all $n$, for all inputs of length $n$ and for all histories, $N$ halts in $T(n)$ time
Definition: $\text{NTIME}(t(n)) =$

$\{ L | L \text{ is decided by a } O(t(n)) \text{ time nondeterministic Turing machine} \}$

$\text{TIME}(t(n)) \subseteq \text{NTIME}(t(n))$

Is $\text{TIME}(t(n)) = \text{NTIME}(t(n))$ for all $t(n)$?
What problems can we efficiently solve nondeterministically, but not deterministically?
The Clique Problem

$k$-clique = complete subgraph on $k$ nodes
The Clique Problem

Find a clique of 1 million nodes?
Assume a reasonable encoding of graphs (example: the adjacency matrix is reasonable)

$\text{CLIQUE} = \{ (G,k) \mid G \text{ is an undirected graph with a } k\text{-clique} \} \]

Theorem: $\text{CLIQUE} \in \text{NTIME}(n^c)$ for some $c > 1$

$N((V,E),k)$:
Nondeterministically guess a subset $S$ of $V$ with $|S| = k$
For all $u, v$ in $S$, if $(u,v)$ is not in $E$ then reject
Accept
The Hamiltonian Path Problem

A Hamiltonian path traverses through each node exactly once
HAMPATH = \{ (G,s,t) \mid G \text{ is a directed graph with a Hamiltonian path from } s \text{ to } t \} \\

Theorem: HAMPATH ∈ NTIME(n^c) for some c > 1 \\

N((V,E),s,t): Nondeterministically guess a sequence v_1, ..., v_{|V|} of vertices \\
If v_i = v_j for some i ≠ j, reject \\
For all i = 1,...,|V|-1, \\
if (v_i,v_{i+1}) \text{ is not in } E \text{ then reject} \\
If (v_1 = s \& v_n = t) \text{ then accept else reject}
Nondeterministic Polynomial Time

\[ \text{NP} = \bigcup_{k \in \mathbb{N}} \text{NTIME}(n^k) \]
Theorem: \( L \in NP \iff \text{There is a constant } k \text{ and polynomial-time TM } V \text{ such that} \)

\[ L = \{ x \mid \exists y \in \Sigma^* \ [ |y| \leq |x|^k \text{ and } V(x,y) \text{ accepts} \} \]

Proof:
1. If \( L = \{ x \mid \exists y \ |y| \leq |x|^k \text{ and } V(x,y) \text{ accepts} \} \) then \( L \in NP \)

Define the NTM \( N(x) \):
- Guess \( y \) of length at most \( |x|^k \)
- Run \( V(x,y) \) and output answer

Then, \( L(N) \) is the set of \( x \) s.t. \( |y| \leq |x|^k \& V(x,y) \text{ accepts} \)

(2) If \( L \in NP \) then \( L = \{ x \mid \exists y \ |y| \leq |x|^k \text{ and } V(x,y) \text{ accepts} \} \)

Suppose \( N \) is a poly-time NTM that decides \( L \). Define \( V(x,y) \) to accept iff \( y \) encodes an accepting computation history of \( N \) on \( x \)
A language $L$ is in NP if and only if there are polynomial-length proofs (aka. certificates or witnesses) for membership in $L$. 

$\text{CLIQUE} = \{(G,k) \mid \exists \text{ subset of nodes } S \text{ such that } S \text{ is a } k\text{-clique in } G \}$

$\text{HAMPATH} = \{(G,s,t) \mid \exists \text{ Hamiltonian path in graph } G \text{ from node } s \text{ to node } t \}$
Boolean Formula Satisfiability

\[ \phi = (\neg x \land y) \lor z \]

logical operations
parentheses

\neg \text{ recedes}
\land \text{ precedes } \lor

Boolean variables (0 or 1)
A *satisfying assignment* is a setting of the variables that makes the formula true.

\[ \phi = \neg(x \land y) \lor z \]

\( x = 1, \ y = 1, \ z = 1 \) is a satisfying assignment for \( \phi \)

(in fact, any assignment with \( z = 1 \) is satisfying)

\[ \phi = \neg(x \lor y) \land (z \land \neg x) \]

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
<th>( z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
A Boolean formula is **satisfiable** if there is a true/false setting to the variables that makes the formula true.

**YES** \[ a \land b \land c \land \neg d \]

**NO** \[ \neg(x \lor y) \land x \]

\[ SAT = \{ \phi \mid \phi \text{ is a satisfiable Boolean formula} \} \]
A 3cnf-formula has the form:

\[(x_1 \lor \neg x_2 \lor x_3) \land (x_4 \lor x_2 \lor x_5) \land (x_3 \lor \neg x_2 \lor \neg x_1)\]

3SAT = \{ \phi | \phi \text{ is a satisfiable 3cnf-formula} \}
Theorem: 3SAT ∈ NP

We can express 3SAT as

3SAT = { φ | φ is in 3cnf and ∃ string y that encodes a satisfying assignment to φ }

The number of variables of φ is at most |φ|, so |y| ≤ |φ|.

Then, argue that the language

3SAT-CHECK = { (φ, y) | φ is in 3cnf and y is a satisfying assignment to φ }

is in P.

(Similarly, SAT ∈ NP)
NP = Problems with the property that, once you have the solution, it is “easy” to verify the solution

When $\phi \in \text{SAT}$, 
or $(G, k) \in \text{CLIQUE}$, 
or $(G, s, t) \in \text{HAMPATH}$,

Can prove that with a short proof that can easily been verified

What if $\phi \notin \text{SAT}$? $(G, k) \notin \text{CLIQUE}$? Or $(G, s, t) \notin \text{HAMPATH}$?
P = the problems that can be efficiently solved

NP = the problems where proposed solutions can be efficiently verified

Is P = NP? can problem solving be automated?

Clay Math Institute in the year 2000: “millennium problems”
If $P = NP$:

Mathematicians may be out of a job

Cryptography as we know it may be impossible

In principle, every aspect of life could be efficiently and globally optimized ... life as we know it would be different!

Conjecture: $P \neq NP$
Parting thoughts:
Polynomial Time: Verifying vs. Deciding