P - class of problems with poly-time solvers.

NP - " " " " " " verifiers.

NPC - " " " " X, such that:

1) X is in NP (poly-time verifiable).
2) X is at least as hard as every problem in NP.
   (For all Z in NP, Z ≤p X)

\[ X \text{ is NPC if } X \text{ is in:} \]
\[ \begin{array}{c}
P \\
NP \\
NPC \\
NP\text{-hard}
\end{array} \]

To prove that a problem X is NPC:
1) Show that there is a poly-time verifier for X.
2) Choose a similar problem Y that is in NPC.
   * Show that Y ≤p X.

\[ \text{\emph{Know}: for all } Z \text{ in NP, } Z \leq p Y \]
\[ \text{If } Y \leq p X \]
\[ \text{Implies } Z \leq p X \]

In words: If we could solve X, we could solve every problem in NP.

\text{contra-positive: If we can't solve every problem in NP, we can't solve X.}

(Most likely) can't solve every problem in NP (unless P = NP).

\[ \therefore \text{ can't solve X.} \]
To show \( Y \in P \):

Recall squaring vs multiplying example:

\[
\begin{align*}
(Y) & \quad (X) \\
\text{Squaring} & \quad \text{Multiplying}
\end{align*}
\]

"If we have an algorithm for multiplying, we can use it for squaring.

How?

ex: \( \text{square}(a) \) ? multiplies 2 numbers

\[
\begin{array}{c}
a \rightarrow (a, a) \rightarrow \text{MULT2} \rightarrow \text{sol}
\end{array}
\]

In general:

\[
\begin{array}{c}
\text{input/instance (Y)} \rightarrow \text{input/instance (X)} \rightarrow \text{ALG(X)} \rightarrow \text{Soln Converter} \rightarrow \text{Soln (Y)}
\end{array}
\]

"(in poly-time)" \( \quad "(\text{in poly-time})"

Note: we just need to show the instance + solution converters!
First NPC Proof: Traveling Salesperson Problem (TSP)

definition: A graph is complete if there is an edge between every pair of vertices.

not complete complete complete

TSP:
Given: (1) a complete graph $G = (V,E)$ where each edge $(uw) \in E$ has cost $c(u,v) > 0$ and
(2) a bound $k$

Goal: Determine whether there is a tour (path that visits every vertex exactly once and returns to origin) with total cost $\leq k$. $\Rightarrow$ Decision question: answer is yes/no

ex: $k = 15$

Yes: $d - a - b - c - d$

\[
\begin{array}{c|cccc}
& a & b & c & d \\
\hline
1 & - & 2 & 6 & 4 \\
2 & 6 & - & 3 & 5 \\
3 & 4 & 3 & - & \\
4 & 5 & 2 & 1 & - \\
\end{array}
\]

Thm: TSP is NP-complete
Proof: (1) TSP $\in$ NP
solution/certificate is a path $S = v_1, v_2, \ldots, v_n v_1$
To verify certificate, check:

1. every vertex (except $v_i$) appears exactly once. $(S = V - v_i)$
2. $v_i$ appears as first & last vertex in $S$
3. sum of edge costs $c_{v_i, v_j}$ in $S \leq k$

Can be checked in poly-time.

(2) TSP is NP-hard.

For all $Z$ in NP, $Z \leq_p X$

Reduction from Ham-Cycle

Ham-Cycle - in a general graph is there a cycle that visits every vertex exactly once?

Will show Ham-Cycle $\leq_p$ TSP.

If we could solve TSP, we could solve Ham-Cycle.

Contrapositive: If we can't solve Ham-Cycle, can't solve TSP.