Recall: \textit{polynomial reductions.}

\[ X \leq_p Y \limp \text{polytime} \quad a \in \text{input of } Y \limp \text{solution to } X \]

e.g., \textsc{CLIQUE} \quad \varnothing \quad \text{No}

- \text{\textsc{SAT} \leq_p \textsc{3SAT}}
- \text{\textsc{SAT} \leq_p \textsc{SAT}}
- \text{\textsc{MIS} \leq_p \textsc{CLIQUE}}

\textbf{NP:} \quad \text{"no polynomials"} \quad \text{not polynomial-time.}

- decision problem \textbf{NP:} \quad \text{solution can be checked in polynomial time.}

Some problems have efficiently checkable "proofs" or "certificates" for \textsc{YES} answers.

\textbf{COMPOSITE:}
\begin{itemize}
  \item given number \( X \), \textit{is this composite?}
  \item non-trivial factors \( p | X \), where \( 1 < p < X \)
\end{itemize}

\textit{Proof:} divide \( X \) by \( p \), and check remainder.

\textit{Check:} divide \( X \) by \( p \), and check remainder.

\textbf{IS:} \quad \textit{is there a \( k \) in set?}

\textit{Proof:} \textit{indeed there is a \( k \)}

\textit{Check:} \textit{check 1 \( \leq k \)}

\[ \text{if } k \leq u, v \leq k \]

https://onedrive.live.com/redir?resid=D2ECBD89F9D7F77E%211110&page=Edit&wd=target%28ECE374 S20%2F Lecture scribbles.on...
\[ P \leq \text{NP} \quad \text{NP} \leq \text{Exp} \]

**NP**: non-deterministic polynomial time

**NP**: non-deterministic polynomial time

\[ \text{NP} \ni \text{P} \quad \text{NP} \ni \text{CNP} \quad \text{CNP} \ni \text{NP} \]

\[ \text{EXP} : \text{exponential time solvable} \]

\[ \text{PRIME} \Rightarrow \text{CNP} \]

\[ \text{NP} \leq \text{EXP} ? \]

In other words, any problem \( X \) that is \( \text{YES} \)

- instances can be checked in \( \text{poly} \) time
- then it can be solved \( \text{both YES \& NO} \) in \( \text{Exp} \)

\[ \text{X} \text{\in} \text{NP} \text{ means:} \]

\[ \exists \text{ C_p (I_x proof)} \]
\[ C_X \text{ was polytime} \iff \exists \text{ proof, so } C_X(I_X, \text{proof}) = 1 \implies I_X \text{ is c YES instance} \]

\[ \forall x \in \text{Exp. time} : C_X(I_x) : \]

\[ i = 0 \]

\[ \text{For i counting sum 0 to p(n)} : \]

\[ \text{search proofs that are i bits long} : \]

\[ \text{Exp time} \leq \text{search proofs that are i bits long} : \]

\[ \text{if } C_X(I_X, \text{proof}) = 1, \text{ return YES} \]

\[ \text{return NO.} \]

\[ \text{Decidable} \]

\[ \text{NP-hard, NP-complete} \]

\[ Y \text{ is NP-hard iff it's at least as hard} \]

\[ \forall X \in \text{NP}, X \leq_Y Y \]

\[ \text{equiv. a solution to } X \implies \text{solution to any NP problem.} \]

\[ \text{NP-complete} : \text{NP-hard and also } \in \text{NP} \]

\[ \text{NP-hard} \]

\[ \text{NP-complete} \]

\[ \text{...} \]

\[ \text{NP-hard, NP-complete} \]

\[ \text{NP-hard} \]
\[
P \neq NP \Rightarrow \text{There are problems that are hard to solve but easy to check.}
\]
\[
P = NP \Rightarrow \text{all easy to check problems are also easy to solve.}
\]
\[
\text{eg, all composite numbers can be checked} \Rightarrow \text{no cryptography could work.}
\]
Usually we assume \( P \neq NP \).

Proving problems are \( NP \)-hard is \( NP \)-complete.
- To show \( \in NP \), prove yes instances are easy to check.
- To show \( \in NP \)-hard:
  1. **Cook-Levin Theorem. Direct way.**
      \[
      \forall X \in NP, \text{we can encode } X \text{ into SAT:}
      \]
      \[
      X \in NP \Rightarrow \exists X (I_X, \text{proof})
      \]
      \[
      \text{where } I_X \text{ is the } \text{in} \text{mediate } \text{w} \text{f } \text{e } \text{val}.
      \]
      \[
      \text{CNF } \text{O} \text{as } 1 (X_1 \lor X_2 \lor X_3)
      \]
      \[
      \land \land \land \land \land \land
      \]
      \[
      \left\{_{\text{3CNF}} \right.
      \]
  2. Show that some \( NP \)-hard problem reduces \( y \).
     \[
     \text{e.g. } 3\text{SAT} \leq_p y.
     \]
     \[
     \Rightarrow \forall X, X \in SAT \leq 3\text{SAT} \leq_p y
     \]
Skill: prove a problem is NP-hard by reducing from a known NP-hard problem.

Ex. \[3\text{SAT} \leq_p \text{k-IS}\] each clause has 3 literals.

\[(x_1 \lor \neg x_2 \lor x_3)\]

Show "encode any instance of 3SAT into k-IS."

- A k-IS must select exactly 1
- A k-IS selects at most 1 from each triangle

Show transformation is polytime.

Good instances of 3SAT \(\Rightarrow\) good instances of k-IS

Good instances of k-IS \(\Rightarrow\) good instances of 3SAT

or Bad instances of 3SAT \(\Rightarrow\) bad instances of k-IS.