Polynomial-time reductions

Suppose $Y$ in P. What else is in P?

**Reduction.** Problem $X$ polynomial-time (Cook) reduces to problem $Y$ if arbitrary instances of problem $X$ can be solved using:

- Polynomial number of standard computational steps, plus
- Polynomial number of calls to oracle that solves problem $Y$.
Polynomial-time reductions

Suppose \( Y \) in P. What else is in P?

**Reduction.** Problem \( X \) polynomial-time (Cook) reduces to problem \( Y \) if arbitrary instances of problem \( X \) can be solved using:

- Polynomial number of standard computational steps, plus
- Polynomial number of calls to oracle that solves problem \( Y \).

**Notation.** \( X \leq_p Y \).

**Note.** We pay for time to write down instances sent to oracle \( \Rightarrow \) instances of \( Y \) must be of polynomial size.

**Caveat.** Don't mistake \( X \leq_p Y \) with \( Y \leq_p X \).
Polynomial-time reductions

**Design algorithms.** If $X \leq_p Y$ and $Y$ can be solved in polynomial time, then $X$ can be solved in polynomial time.

**Establish intractability.** If $X \leq_p Y$ and $X$ cannot be solved in polynomial time, then $Y$ cannot be solved in polynomial time.

**Establish equivalence.** If both $X \leq_p Y$ and $Y \leq_p X$, we use notation $X \equiv_p Y$. In this case, $X$ can be solved in polynomial time iff $Y$ can be.

**Bottom line.** Reductions classify problems according to relative difficulty.
Independent set

**INDEPENDENT-SET.** Given graph $G = (V, E)$ and integer $k$, is there subset $S \subseteq V$, with $|S| \geq k$, s.t. no edge contained in $S$?

**Ex.** Is there an independent set of size $\geq 6$?  
**Ex.** Is there an independent set of size $\geq 7$?

[Diagram of a graph with independent set highlighted]
**Vertex cover**

**VERTEX-COVER.** Given graph $G = (V, E)$ and integer $k$, is there $S \subseteq V$ with $|S| \leq k$, s.t. each edge touches $S$?

**Ex.** Is there a vertex cover of size $\leq 4$?
**Ex.** Is there a vertex cover of size $\leq 3$?
Vertex cover and independent set reduce to one another

**Theorem.** $\text{VERTEX-COVER} \equiv_p \text{INDEPENDENT-SET}.$

**Pf.** We show $S$ is an independent set of size $k$ iff $V - S$ is a vertex cover of size $n - k.$
**Theorem.** \textsc{Vertex-Cover} $\equiv_{p} \textsc{Independent-Set}$.

**Pf.** We show $S$ is an independent set of size $k$ iff $V - S$ is a vertex cover of size $n - k$.

$\Rightarrow$

- Let $S$ be independent set.
- Consider edge $\{u, v\}$.
- $S$ independent $\Rightarrow$ either $u \notin S$ or $v \notin S$ (or both)
  $\Rightarrow$ either $u \in V - S$ or $v \in V - S$ (or both).
- Thus, $V - S$ covers $\{u, v\}$. 
Vertex cover and independent set reduce to one another

**Theorem.** \textsc{Vertex-Cover} $\equiv_P$ \textsc{Independent-Set}.

**Pf.** We show $S$ is an independent set of size $k$ iff $V - S$ is a vertex cover of size $n - k$.

$\Leftarrow$

- Let $V - S$ be vertex cover.
- Consider two nodes $u \in S$ and $v \in S$.
- $\{u, v\} \notin E$ since $V - S$ is a vertex cover $\Rightarrow S$ independent set. $\blacksquare$
**Set cover**

**Set-Cover.** Given a collection $S_1, S_2, \ldots, S_m$ of subsets of $\{1,2,\ldots,n\}$, and an integer $k$, does there exist $\leq k$ of these sets whose union is equal to $U$?

**Sample application.**

- $m$ available pieces of software.
- Set of $n$ capabilities that we would like our system to have.
- The $i^{th}$ piece of software provides the set $S_i \subseteq U$ of capabilities.
- Goal: achieve all $n$ capabilities using fewest pieces of software.

\[
\begin{align*}
U &= \{ 1, 2, 3, 4, 5, 6, 7 \} \\
S_1 &= \{ 3, 7 \} \quad S_4 = \{ 2, 4 \} \\
S_2 &= \{ 3, 4, 5, 6 \} \quad S_5 = \{ 5 \} \\
S_3 &= \{ 1 \} \quad S_6 = \{ 1, 2, 6, 7 \} \\
k &= 2
\end{align*}
\]

a set cover instance
Vertex cover reduces to set cover

**Theorem.** VERTEX-COVER $\leq_p$ SET-COVER.

**Pf.** Given VERTEX-COVER instance $G = (V, E)$, we construct a SET-COVER instance that has a set cover of size $k$ iff $G$ has a vertex cover of size $k$.

**Construction.**

- Universe $= E$.
- Include one set for each node $v \in V$: $S_v = \{ e \in E : e \text{ incident to } v \}$.

**Example:**

- Let $U = \{ 1, 2, 3, 4, 5, 6, 7 \}$
- Let $S_a = \{ 3, 7 \}$, $S_b = \{ 2, 4 \}$, $S_c = \{ 3, 4, 5, 6 \}$, $S_d = \{ 5 \}$, $S_e = \{ 1 \}$, $S_f = \{ 1, 2, 6, 7 \}$

**Vertex cover instance** (k = 2)

**Set cover instance** (k = 2)
Lemma. \( G = (V, E) \) contains a vertex cover of size \( k \) iff \( (U, S) \) contains a set cover of size \( k \).

\[ \text{Pf. } \Rightarrow \text{ Let } X \subseteq V \text{ be a vertex cover of size } k \text{ in } G. \]

- Then \( Y = \{ S_v : v \in X \} \) is a set cover of size \( k \).

Vertex cover reduces to set cover

<table>
<thead>
<tr>
<th>vertex cover instance</th>
<th>set cover instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k = 2 )</td>
<td>( k = 2 )</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
U &= \{ 1, 2, 3, 4, 5, 6, 7 \} \\
S_a &= \{ 3, 7 \} & S_b &= \{ 2, 4 \} \\
\text{**} S_c &= \{ 3, 4, 5, 6 \} & S_d &= \{ 5 \} \\
S_e &= \{ 1 \} & S_f &= \{ 1, 2, 6, 7 \}
\end{align*}
\]
Vertex cover reduces to set cover

Lemma. $G = (V, E)$ contains a vertex cover of size $k$ iff $(U, S)$ contains a set cover of size $k$.

Pf. ⇐ Let $Y \subseteq S$ be a set cover of size $k$ in $(U, S)$.
    - Then $X = \{ v : S_v \in Y \}$ is a vertex cover of size $k$ in $G$. ■
Satisfiability

Literal. A boolean variable or its negation.

Clause. A disjunction of literals.

Conjunctive normal form. A propositional formula \( \Phi \) that is the conjunction of clauses.

\textbf{SAT}. Given CNF formula \( \Phi \), does it have a satisfying truth assignment?

\textbf{3-SAT}. SAT where each clause contains exactly 3 literals (and each literal corresponds to a different variable).

\begin{center}
\begin{figure}
\centering
\includegraphics[width=\textwidth]{sat.png}
\end{figure}
\end{center}

\textit{yes instance:} \( x_1 = \text{true}, x_2 = \text{true}, x_3 = \text{false}, x_4 = \text{false} \)

Key application. Electronic design automation (EDA).
3-satisfiability reduces to independent set

**Theorem.** $3$-SAT $\leq_P$ INDEPENDENT-SET.

**Pf.** Given an instance $\Phi$ of 3-SAT, we construct an instance $(G, k)$ of INDEPENDENT-SET that has an independent set of size $k$ iff $\Phi$ is satisfiable.

**Construction.**
- $G$ contains 3 nodes for each clause, one for each literal.
- Connect 3 literals in a clause in a triangle.
- Connect literal to each of its negations.

$k = 3$
3-satisfiability reduces to independent set

**Lemma.** $G$ contains independent set of size $k = |\Phi|$ iff $\Phi$ is satisfiable.

**Pf.** $\Rightarrow$ Let $S$ be independent set of size $k$.
- $S$ must contain exactly one node in each triangle.
- Set these literals to true (and remaining variables consistently).
- Truth assignment is consistent and all clauses are satisfied.

**Pf** $\Leftarrow$ Given satisfying assignment, select one true literal from each triangle. This is an independent set of size $k$. □
3-colorability

3-COLOR. Given an undirected graph $G$, can the nodes be colored red, green, and blue so that no adjacent nodes have the same color?
3-satisfiability reduces to 3-colorability

**Theorem.** $3\text{-SAT} \leq_p 3\text{-COLOR}$.

**Pf.** Given $3\text{-SAT}$ instance $\Phi$, we construct an instance of $3\text{-COLOR}$ that is 3-colorable iff $\Phi$ is satisfiable.
3-satisfiability reduces to 3-colorability

Construction.

(i) Create a graph $G$ with a node for each literal.
(ii) Connect each literal to its negation.
(iii) Create 3 new nodes $T$, $F$, and $B$; connect them in a triangle.
(iv) Connect each literal to $B$.
(v) For each clause $C_j$, add a gadget of 6 nodes and 13 edges.

true

false

T

F

base

B

$x_1$ $x_1$ $x_2$ $x_2$ $x_3$ $x_3$ $x_n$ $x_n$
3-satisfiability reduces to 3-colorability

**Lemma.** Graph $G$ is 3-colorable iff $\Phi$ is satisfiable.

**Pf.** Suppose graph $G$ is 3-colorable.

- Consider assignment that sets all $T$ literals to true.
- (iv) ensures each literal is $T$ or $F$.
- (ii) ensures a literal and its negation are opposites.
Lemma. Graph $G$ is 3-colorable iff $\Phi$ is satisfiable.

Pf. Suppose graph $G$ is 3-colorable.

- Consider assignment that sets all $T$ literals to true.
- (iv) ensures each literal is $T$ or $F$.
- (ii) ensures a literal and its negation are opposites.
- (v) ensures at least one literal in each clause is $T$. 

![Diagram of 6-node gadget](image-url)
3-satisfiability reduces to 3-colorability

**Lemma.** Graph $G$ is 3-colorable iff $\Phi$ is satisfiable.

**Pf.** $\Rightarrow$ Suppose graph $G$ is 3-colorable.

- Consider assignment that sets all $T$ literals to true.
- (iv) ensures each literal is $T$ or $F$.
- (ii) ensures a literal and its negation are opposites.
- (v) ensures at least one literal in each clause is $T$.
3-satisfiability reduces to directed hamilton cycle

**DIR-HAM-CYCLE:** Given a digraph $G = (V, E)$, does there exist a simple directed cycle $\Gamma$ that contains every node in $V$?

**Theorem.** $3$-SAT $\leq_p$ DIR-HAM-CYCLE.

**Pf.** Given an instance $\Phi$ of $3$-SAT, we construct an instance of DIR-HAM-CYCLE that has a Hamilton cycle iff $\Phi$ is satisfiable.

**Construction.** First, create graph that has $2^n$ Hamilton cycles which correspond in a natural way to $2^n$ possible truth assignments.
3-satisfiability reduces to directed hamilton cycle

**Construction.** Given 3-SAT instance $\Phi$ with $n$ variables $x_i$ and $k$ clauses.

- Construct $G$ to have $2^n$ Hamilton cycles.
- Intuition: traverse path $i$ from left to right $\iff$ set variable $x_i = true$. 

![Diagram of graph](image-url)
3-satisfiability reduces to directed hamilton cycle

**Construction.** Given 3-SAT instance $\Phi$ with $n$ variables $x_i$ and $k$ clauses.

- For each clause, add a node and 6 edges.

\[ C_1 = x_1 \lor \overline{x_2} \lor x_3 \]  
\[ C_2 = \overline{x_1} \lor \overline{x_2} \lor \overline{x_3} \]
3-satisfiability reduces to directed hamilton cycle

Lemma. \( \Phi \) is satisfiable iff \( G \) has a Hamilton cycle.

Pf. \( \Rightarrow \)

• Suppose 3-SAT instance has satisfying assignment \( x^* \).
• Then, define Hamilton cycle in \( G \) as follows:
  - if \( x^*_i = \text{true} \), traverse row \( i \) from left to right
  - if \( x^*_i = \text{false} \), traverse row \( i \) from right to left
  - for each clause \( C_j \), there will be at least one row \( i \) in which we are going in
    "correct" direction to splice clause node \( C_j \) into cycle
    (and we splice in \( C_j \) exactly once)
3-satisfiability reduces to directed hamilton cycle

**Lemma.** \( \Phi \) is satisfiable iff \( G \) has a Hamilton cycle.

**Pf.** \( \iff \)

- Suppose \( G \) has a Hamilton cycle \( \Gamma \).
- If \( \Gamma \) enters clause node \( C_j \), it must depart on mate edge.
  - nodes immediately before and after \( C_j \) are connected by an edge \( e \in E \)
  - removing \( C_j \) from cycle, and replacing it with edge \( e \) yields Hamilton cycle on \( G - \{ C_j \} \)
- Continuing in this way, we are left with a Hamilton cycle \( \Gamma' \) in \( G - \{ C_1, C_2, \ldots, C_k \} \).
- Set \( x^*_{i} = true \) iff \( \Gamma' \) traverses row \( i \) left to right.
- Since \( \Gamma \) visits each clause node \( C_j \), at least one of the paths is traversed in "correct" direction, and each clause is satisfied. \( \blacksquare \)