Polynomial Time Reduction

- Polynomial Time Reduction Definition
- Reduction by Equivalence
- Reduction from Special Cases to General Case
- Reduction by Encoding with Gadgets
- Transitivity of Reductions
- Decision, Search and Optimization Problem
- Self-Reducibility

Imdad ullah Khan

Polynomial time reduction is a way to compare hardness of problems

Problem A is polynomial time reducible to Problem B, $A \leq_p B$

If any instance of problem A can be solved using a polynomial amount of computation plus a polynomial number of calls to a solution of problem B

If any algorithm for problem B can be used [called (once or more) with *'clever'* legal inputs] to solve any instance of problem A

Subroutine for B takes an instance y of B and returns the solution $B(y)$

Algorithm for A transforms an instance x of A to an instance y of B. Then transforms $B(y)$ to $A(x)$

Reduction by encoding with gadgets

$$3\text{-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

$$f = (x_{11} \vee x_{12} \vee x_{13}) \wedge (x_{21} \vee x_{22} \vee x_{23}) \wedge \ldots \quad \ldots \wedge (x_{m1} \vee x_{m2} \vee x_{m3})$$

We need to set each of $x_1, \ldots, x_n$ to $0/1$ so as $f = 1$

Alternatively,

1. We need to pick a literal from each clause and set it to 1
2. But we cannot make conflicting settings

Reduction by encoding with gadgets

$$3\text{-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For clauses with 2 and 1 literal make an edge or a node
- Make edges between literals appearing in different clauses as complements

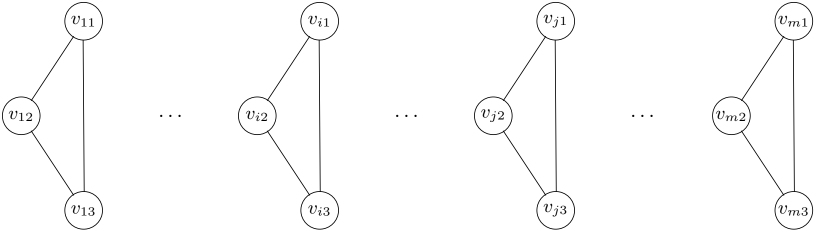
$$(x_{11} \vee x_{12} \vee x_{13}) \wedge \ \ldots \ \wedge (x_{i1} \vee x_{i2} \vee x_{i3}) \wedge \ \ldots \ \wedge (x_{j1} \vee x_{j2} \vee x_{j3}) \wedge \ \ldots \ \wedge (x_{m1} \vee x_{m2} \vee x_{m3})$$

Reduction by encoding with gadgets

$$3\text{-}\textsc{sat}(f) \quad \leq_p \quad \textsc{independent-set}(G,k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For clauses with 2 and 1 literal make an edge or a node
- Make edges between literals appearing in different clauses as complements

$$(x_{11} \vee x_{12} \vee x_{13}) \wedge \ \ldots \ \wedge (x_{i1} \vee x_{i2} \vee x_{i3}) \wedge \ \ldots \ \wedge (x_{j1} \vee x_{j2} \vee x_{j3}) \wedge \ \ldots \ \wedge (x_{m1} \vee x_{m2} \vee x_{m3})$$

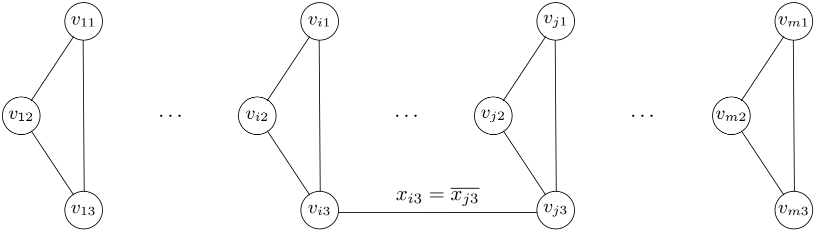


Reduction by encoding with gadgets

$$\text{3-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For clauses with 2 and 1 literal make an edge or a node
- Make edges between literals appearing in different clauses as complements

$$(x_{11} \vee x_{12} \vee x_{13}) \wedge \ \ldots \ \wedge (x_{i1} \vee x_{i2} \vee x_{i3}) \wedge \ \ldots \ \wedge (x_{j1} \vee x_{j2} \vee x_{j3}) \wedge \ \ldots \ \wedge (x_{m1} \vee x_{m2} \vee x_{m3})$$



Reduction by encoding with gadgets

$3\text{-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For clauses with 2 and 1 literal make an edge or a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee \overline{x_3}) \quad \wedge \quad (\overline{x_1} \vee \overline{x_3} \vee x_4) \quad \wedge \quad (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$$

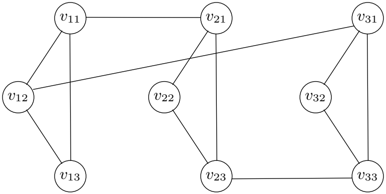
Reduction by encoding with gadgets

$$\text{3-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For smaller clauses make an edge or just a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee \overline{x_3}) \quad \wedge \quad (\overline{x_1} \vee \overline{x_3} \vee x_4) \quad \wedge \quad (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$$



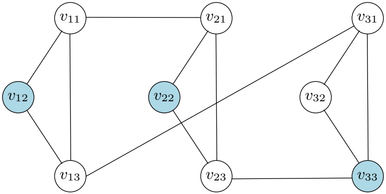
Reduction by encoding with gadgets

$$\text{3-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For smaller clauses make an edge or just a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee \overline{x_3}) \quad \wedge \quad (\overline{x_1} \vee \overline{x_3} \vee x_4) \quad \wedge \quad (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$$



$$x_1 = 1, \overline{x_3} = 1, \overline{x_4} = 1$$

Reduction by encoding with gadgets

$$3\text{-}\textsc{sat}(f) \quad \leq_p \quad \textsc{independent-set}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For smaller clauses make an edge or just a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$$

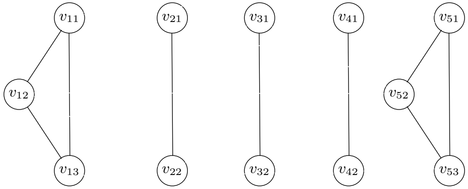
Reduction by encoding with gadgets

$$3\text{-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For smaller clauses make an edge or just a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$$



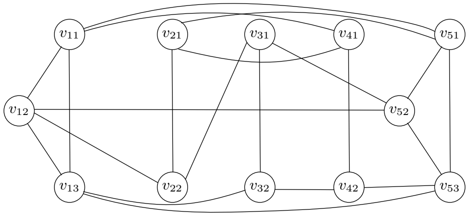
Reduction by encoding with gadgets

$$\text{3-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For smaller clauses make an edge or just a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$$



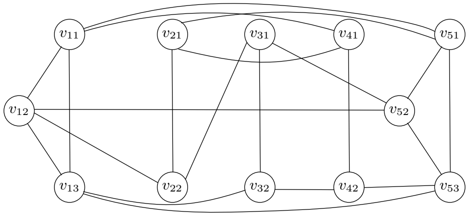
Reduction by encoding with gadgets

$$3\text{-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

- Given f on n variables and m clauses - Make a graph G
- For each clause make a triangle with nodes labeled with literals
- For smaller clauses make an edge or just a node
- Make edges between literals appearing in different clauses as complements

Theorem: f is satisfiable iff G has an independent set of size m

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$$



No satisfying assignmnet, No independent set of size 5

Reduction by encoding with gadgets

$$3\text{-SAT}(f) \quad \leq_p \quad \text{INDEPENDENT-SET}(G, k)$$

Theorem: f is satisfiable iff G has an independent set of size m

The reduction is as follows:

- Let $\mathcal{A}$ be an algorithm for the INDEPENDENT-SET(G, k) problem
- We will use $\mathcal{A}$ to solve the 3-SAT(f) problem
- Given any instance f of 3-SAT(f) on n variables and m clauses
- Construct the graph as outlined above
- Call $\mathcal{A}$ on $[G, m]$
- if $\mathcal{A}$ returns **Yes**, declare f satisfiable and vice-versa
- G can be constructed in time polynomial in n and m
- Hence, this is a polynomial time reduction

Reduction by encoding with gadgets

$$\text{SAT}(f) \quad \leq_p \quad 3\text{-SAT}(f')$$

- Given a CNF formula f on variables $X = \{x_1, \ldots, x_n\}$, D: new variables
- Construct an equivalent 3-CNF formula f' on variables $X \cup \{d_1, d_2, \ldots\}$
- Initialize $f' = f$. For a long clause $C = (x_{i1} \vee x_{i2} \vee x_{i3} \vee x_{i4} \vee \ldots)$ in f'
- Add the clauses $(x_{i1} \vee x_{i2} \vee d_i) \wedge (\overline{d_i} \vee x_{i3} \vee x_{i4} \vee \ldots)$ to f'
- The new (long clause) is shorter than C

$$(x_{i1} \vee x_{i2} \vee \underbrace{x_{i3} \vee x_{i4} \vee \ldots}_{y}) \iff (x_{i1} \vee x_{i2} \vee d_i) \wedge (\overline{d_i} \vee \underbrace{x_{i3} \vee x_{i4} \vee \ldots}_{y})$$

- Suppose $(x_{i1} \vee x_{i2} \vee \underbrace{x_{i3} \vee x_{i4} \vee \ldots}_{y})$ is satisfiable
- If $x_{i1} \vee x_{i2} = 1$. Set $d_i = 0$ ▷ RHS is also satisfiable
- If $x_{i1} \vee x_{i2} = 0$, then $y = 1$. Set $d_i = 1$ ▷ RHS is also satisfiable

Reduction by encoding with gadgets

$$\textsc{sat}(f) \;\; \leq_p \;\; 3\text{-}\textsc{sat}(f')$$

- Given a CNF formula f on variables $X = \{x_1, \ldots, x_n\}$, D: new variables
- Construct an equivalent 3-CNF formula f' on variables $X \cup \{d_1, d_2, \ldots\}$
- Initialize $f' = f$. For a long clause $C = (x_{i1} \vee x_{i2} \vee x_{i3} \vee x_{i4} \vee \ldots)$ in f'
- Add the clauses $(x_{i1} \vee x_{i2} \vee d_i) \wedge (\overline{d_i} \vee x_{i3} \vee x_{i4} \vee \ldots)$ to f'
- The new (long clause) is shorter than C

$$(x_{i1} \vee x_{i2} \vee \underbrace{x_{i3} \vee x_{i4} \vee \ldots}_{y}) \Longleftrightarrow (x_{i1} \vee x_{i2} \vee d_i) \wedge (\overline{d_i} \vee \underbrace{x_{i3} \vee x_{i4} \vee \ldots}_{y})$$

- Suppose $(x_{i1} \vee x_{i2} \vee d_i) \wedge (\overline{d_i} \vee \underbrace{x_{i3} \vee x_{i4} \vee \ldots}_{y})$ is satisfiable
- If $d_i = 1$, then $\overline{d_i} = 0$ and $y = 1$ ▷ LHS is also satisfiable
- If $d_i = 0$, then $\overline{d_i} = 1$ and $x_{i1} \vee x_{i2} = 1$ ▷ LHS is also satisfiable

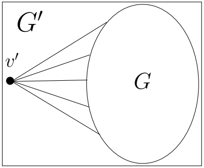
Reduction by encoding with gadgets

$$\text{HAM-PATH}(G) \quad \leq_p \quad \text{HAM-CYCLE}(G)$$

- Let $\mathcal{A}$ be an algorithm for HAM-CYCLE(G)
- Given an instance G of HAM-PATH(G)
- Let G' be G plus a dummy vertex v' adjacent to all vertices in $V(G)$

G' has a Hamiltonian cycle if and only if G has a Hamiltonian path

- Call $\mathcal{A}$ on G'
- If $\mathcal{A}$ outputs **Yes** we will output **Yes** and vice-versa

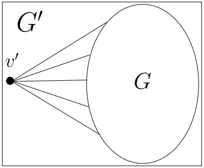


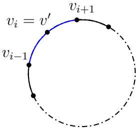
HAM-PATH(G) $\leq_p$ HAM-CYCLE(G)

- Let $\mathcal{A}$ be an algorithm for HAM-CYCLE(G)
- Given an instance G of HAM-PATH(G)
- Let G' be G plus a dummy vertex v' adjacent to all vertices in $V(G)$

G' has a Hamiltonian cycle if and only if G has a Hamiltonian path

- Call $\mathcal{A}$ on G'
- If $\mathcal{A}$ outputs **Yes** we will output **Yes** and vice-versa





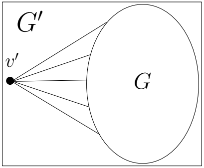
Reduction by encoding with gadgets

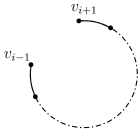
$\textsc{ham-path}(G) \leq_p \textsc{ham-cycle}(G)$

- Let $\mathcal{A}$ be an algorithm for $\textsc{ham-cycle}(G)$
- Given an instance G of $\textsc{ham-path}(G)$
- Let G' be G plus a dummy vertex v' adjacent to all vertices in $V(G)$

G' has a Hamiltonian cycle if and only if G has a Hamiltonian path

- Call $\mathcal{A}$ on G'
- If $\mathcal{A}$ outputs **Yes** we will output **Yes** and vice-versa

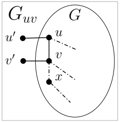


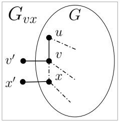


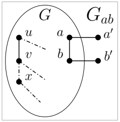
Polynomial Time Reduction: Cook Reducibility

$\text{HAM-CYCLE}(G) \quad \leq_p \quad \text{HAM-PATH}(G)$

- Let $\mathcal{A}$ be an algorithm for $\text{HAM-PATH}(G)$
- Given an instance $G = (V, E)$ of $\text{HAM-CYCLE}(G)$
- For each edge $e = (u, v) \in E(G)$ make the graph $G_e = (V_e, E_e)$
- $V_e = V \cup \{u', v'\}$ and $E_e = E \cup \{(u, u'), (v, v')\}$







G has a Hamiltonian cycle if and only if some G_e has a Hamiltonian path

- Call $\mathcal{A}$ on each of G_{uv} $\triangleright$ $O(|E|)$ calls
- If $\mathcal{A}$ outputs **Yes** on any G_e, we will output **Yes**
- If $\mathcal{A}$ outputs **No** on all G_e, we will output **No**