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 $A(x)$

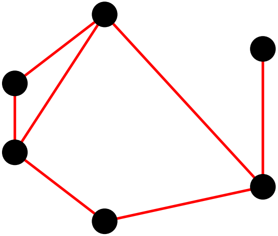
Reduction by (Complementary) Equivalence

Theorem

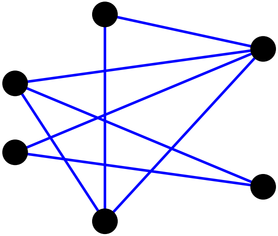
G has an independent set of size k iff $\overline{G}$ has a clique of size k

Recall the complement of a graph $G = (V, E)$ is the graph

$\overline{G} = (V, \overline{E})$, where $\overline{E} = \{(u, v) : (u, v) \notin E\}$



A graph G



$\overline{G}$

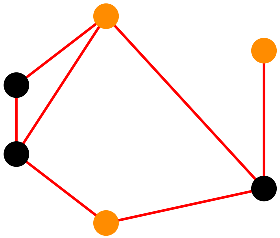
Reduction by (Complementary) Equivalence

Theorem

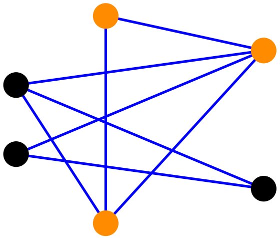
G has an independent set of size k iff $\overline{G}$ has a clique of size k

Recall that for $G = (V, E)$ its complement is a graph

$\overline{G} = (V, \overline{E})$, where $\overline{E} = \{(u, v) : (u, v) \notin E\}$



An independent set of size 3



The same 3 vertices make a clique in $\overline{G}$

Reduction by (Complementary) Equivalence

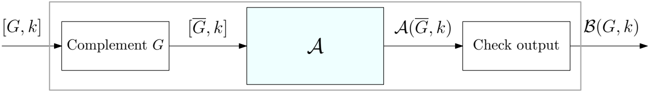
Problem A is polynomial time reducible to Problem B, $A \leq_p B$

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

$$\textsc{clique}(G,k) \quad \leq_p \quad \textsc{ind-set}(G,k)$$

- Let $\mathcal{A}$ be an algorithm solving $\textsc{ind-set}(G,k)$ for any G and $k \in \mathbb{Z}$
- Let $[G,k]$ be an instance of the $\textsc{clique}$ problem
- Compute the complement $\overline{G}$ of G ▷ Polytime
- Call $\mathcal{A}$ on $[\overline{G},k]$
- If it outputs **Yes**, output **Yes** for the problem $\textsc{clique}(G,k)$
- Else output **No**

Algorithm $\mathcal{B}$ takes an instance $[G,k]$ of $\textsc{clique}$ returns **Yes** if G has a clique of size k else returns **No**



Algorithm $\mathcal{B}$ solves $\textsc{clique}$(G,K) problem using a solution $\mathcal{A}$ for $\textsc{independent-set}$ problem

Why Study both CLIQUE or INDEPENDENT-SET?

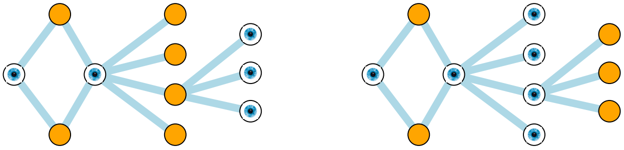
Theorem

G has an independent set of size k iff $\overline{G}$ has a clique of size k

- Given this complementary equivalence should we study both problems?
- Both are "hard" problems
- In practice an approximation algorithm is used for real world graphs
- Most real world graphs are very sparse
- Hence, their complements are very dense
- So applying the same algorithm on the complement will not be as efficient

Reduction by (Complementary) Equivalence

Theorem: $S \subset V$ is an independent in G S iff $V \setminus S$ is a vertex cover in G



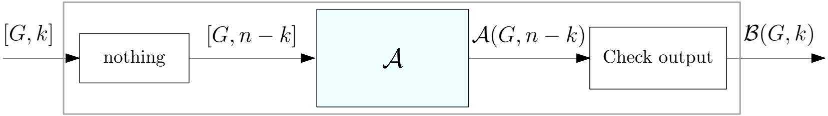
- Let S be an independent set we show that $\overline{S} = V \setminus S$ is a vertex cover
  - For any edge (u, v), either $u \notin S$ or $v \notin S \implies$ either $u \in \overline{S}$ or $v \in \overline{S}$
  - Hence $\overline{S}$ is a vertex cover
- Let C be a vertex cover we show that $\overline{C}$ is an independent set
  - For any edge (u, v) it cannot be that $u \notin C$ and $v \notin C$
  - It cannot be that $u \in \overline{C}$ and $v \in \overline{C}$
  - Hence $\overline{C}$ is an independent set

Reduction by (Complementary) Equivalence

IND-SET$(G, k) \quad \leq_p \quad$ VERTEX-COVER(G, k')

- Let $\mathcal{A}$ be an algorithm solving VERTEX-COVER(G, k) for any G, $k \in \mathbb{Z}$
- Let $[G, t]$ be an instance of the IND-SET problem
- Call $\mathcal{A}$ on $[G, n - t]$
- If it outputs **Yes**, output **Yes** for IND-SET(G, t)
- Else output **No**

$\mathcal{B}$ takes an instance $[G, k]$ of INDEPENDENT-SET returns **YES** if G has an indep.set of size k else returns **NO**



Algorithm $\mathcal{B}$ solves INDEPENDENT-SET(G,K) problem using solution, $\mathcal{A}$ for VERTEX-COVER problem