NP-Complete: Reductions

Definition NP Complete

A problem $Y$ is said to be NP-complete if the following two properties hold:

- $Y \in \text{NP}$
- For all $X \in \text{NP}$, $X \leq_p Y$
NP-Complete: Reductions

Problem: Given that $X$ is NP-complete, prove that $Y$ is NP-complete.

Step 1: $Y \in \text{NP}$

Step 2: $X \leq_p Y$
NP-Complete: Reductions

- Problem: Given that X is NP-complete, prove that Y is NP-complete.

Step 1: \( Y \in \text{NP} \)

**NP** stands for Non-deterministic Polynomial time.
- Solutions are small (== polynomial-size)
- Guessing is free, but solutions must be checkable in polynomial time
  “Succinct Certificate”
NP-Complete: Reductions

**Step 1:** \( Y \in \text{NP} \)

**NP** stands for Non-deterministic Polynomial time.
- Solutions are small (== polynomial-size)
- Guessing is free, but solutions must be checkable in polynomial time
  “Succinct Certificate”

What do we mean when we say a problem is in P?
A solution can be found in polynomial time
What do we mean when we say a problem is in NP?
A solution can be verified in polynomial time
NP-Complete: Reductions

**Problem**: Given that X is NP-complete, prove that Y is NP-complete.

**Step 1**: $Y \in NP$

**Examples**:

- **Vertex Cover**: $VC \in NP$ since we can guess a cover of size $\leq k$ and check it in polynomial time?

- **Independent Set**: $IS \in NP$ since we can guess a set of size $\geq k$ and check it in polynomial time?

- **SAT**: $SAT \in NP$ since we can guess a satisfying assignment and check it in polynomial time?

- **Hamiltonian Cycle**: $HC \in NP$ since we can guess a cycle and check it in polynomial time?
Polynomial-Time Reducibility

Step 2: $X \leq_p Y$

If problem $X$ can be reduced to problem $Y$, we denote this by $X \leq_p Y$.

This means “$X$ is polynomial-time reducible to $Y$.”

If you had a black box that can solve instances of problem $Y$, how can you solve any instance of $X$ using polynomial number of steps, plus a polynomial number of calls to the black box that solves $X$?
NP-Complete: Reductions

**Problem**: Given that $X$ is NP-complete, prove that $Y$ is NP-complete.

**Step 2**: $X \leq_p Y$

If there exists an algorithm for solving $X$ that would be polynomial if we took no account of the time needed to solve arbitrary instances of $Y$. 

---

**Diagram**:

- **Instance of $X = u_1$**
- **Poly-time Transform**
- **Instance of $Y = u_2$**
- **Decider of $Y$**
  - **YES**
  - **NO**
- **Decider of $X$**
  - **YES**
  - **NO**
Problem: Given that Vertex Cover (VC) is NP-complete, prove that Independent Set is NP-complete.

Independent set (IS): Given an undirected graph $G$ and integer $k$, does $G$ have a set of at least $k$ independent vertices?

Vertex Cover (VC): Given an undirected graph $G$ and integer $k$, does $G$ have a vertex cover of size at most $k$. 
Problem: Given that Vertex Cover (VC) is NP-complete, prove that Independent Set (IS) is NP-complete.

Step 1: Independent Set ∈ NP
Step 2: Vertex Cover ≤_p Independent Set

Step 1: The certificate is an independent set of size at least k. The certifier checks if the given set is indeed an independent set of size at least k. (Guessing is free, but solutions must be checkable in polynomial time)

Step 2: We will prove Independent set is NP Complete by reducing Vertex cover to independent set.
Independent set (IS)

Problem: Given that Vertex Cover (VC) is NP-complete, prove that Independent Set (IS) is NP-complete.

Step 2: VC ≤ₚ IS

Goal: Suppose that we have an efficient algorithm for solving Independent Set, it can simply be used to decide whether \( G \) has a vertex cover of size at most \( k \), by asking it to determine whether \( G \) has an independent set of size at least \( n - k \).
Problem: Given that Vertex Cover (VC) is NP-complete, prove that Independent Set is NP-complete.

Step 2: Vertex Cover \( \leq_p \) Independent Set

Goal: Suppose that we have an efficient algorithm for solving Independent Set, it can simply be used to decide whether \( G \) has a vertex cover of size at most \( k \), by asking it to determine whether \( G \) has an independent set of size at least \( n - k \).

If \( G = (V, E) \) is a graph, then \( S \) is an independent set if and only if \( V - S \) is a vertex cover.
Independent set (IS)

Problem: Given that Vertex Cover (VC) is NP-complete, prove that Independent Set is NP-complete.

Step 2: Vertex Cover $\leq_p$ Independent Set

If $G = (V, E)$ is a graph, then $V - S$ is a vertex cover if and only if $S$ is an independent set.

- **If part** - Suppose $S$ is an independent set, and let $e = (u, v)$ be some edge. Only one of $u, v$ can be in $S$. Hence, at least one of $u, v$ is in $V - S$. So, $V - S$ is a vertex cover.

- **Only if part** - Suppose $V - S$ is a vertex cover, and let $u, v \in S$. There can’t be an edge between $u$ and $v$ (otherwise, that edge wouldn’t be covered in $V - S$). So, $S$ is an independent set.
Independent set (IS)

Step 2 : Vertex Cover \( \leq_p \) Independent Set

Proof : Suppose that we have an efficient algorithm for solving Independent Set, it can simply be used to decide whether \( G \) has a vertex cover of size at most \( k \), by asking it to determine whether \( G \) has an independent set of size at least \( n - k \).
Hamiltonian Path (HP)

HAMILTONIAN-CYCLE (HC): Given an undirected graph, determine if there is a Hamiltonian cycle in the graph.

Hamiltonian cycle: A cycle that visits each vertex exactly once.
Hamiltonian Path (HP)

HAMILTONIAN-CYCLE (HC): Given an undirected graph, determine if there is a Hamiltonian cycle in the graph.

Hamiltonian cycle: A cycle that visits each vertex exactly once.

HAMILTONIAN-PATH (HP): Given an undirected graph G, determine if there is a Hamiltonian path in the graph.

Hamiltonian path: A path in any directed graph that visits each vertex exactly once.
Hamiltonian Path (HP)

Problem: Given that Hamiltonian Cycle is NP-complete, prove that Hamiltonian path is NP-complete.

Step 1: HAMILTONIAN-PATH ∈ NP
A Hamiltonian path acts as a certificate. The certifier checks if the given path is a valid path (path in G) and visits each vertex exactly once.

Step 2: HAMILTONIAN-CYCLE ≤_p HAMILTONIAN-PATH
Hamiltonian path (HP)

**Step 2:** $HC \leq_p HP$

- Consider the graph $G'$ constructed from graph $G = (V, E)$. Choose an arbitrary node $v$, and make $v'$ a copy of $v$, and add vertices $s$, $t$ connected to $v$ and $v'$ respectively. That is $G'$ has vertices $V \cup \{v', s, t\}$ and edges $E \cup \{(v', w) | (v, w) \in E\} \cup \{(s', v), (v', t)\}$

**Goal:** Suppose that we have an efficient algorithm for solving Hamiltonian path, it can simply be used to decide whether $G$ has a Hamiltonian cycle, by asking it to determine whether $G'$ has an Hamiltonian path.
Step 2: HC $\leq_p$ HP

There is a Hamiltonian cycle in $G$ if and only there is a Hamiltonian path in $G'$.

- **If part**—If $G'$ has a Hamiltonian Path, then it must have $s$ and $t$ as end points, in which case it will be of form $(s, v), (v, y), edges, (y', v'), (v', t)$. But then $G$ will have a Hamiltonian Cycle $(v, y), edges, (y', v)$.

- **Only if part**—If $G$ has a Hamiltonian Cycle, we may write it as $(v, u), edges', (u', v)$. But then $(s, v), (v, u), edges', (u', v'), (v', t)$ would be a Hamiltonian path in $G'$. 
Hamiltonian path (HP)

Step 2: HAMILTONIAN-CYCLE $\leq_p$ HAMILTONIAN-PATH

Suppose that we have an efficient algorithm for solving Hamiltonian path, it can simply be used to decide whether $G$ has a Hamiltonian cycle, by asking it to determine whether $G'$ has an Hamiltonian path.
Degree bounded Spanning Tree (k-ST)

**Problem:** Given that Hamiltonian-Path is NP-complete, prove that degree-k spanning tree is NP-complete.

**Degree-k spanning tree (k-ST):** A spanning tree of a graph $G = (V, E)$ is a tree containing every vertex in $G$, and whose edges are from $E$. A degree-$k$ spanning tree is a spanning tree such that degree of each node is at most $k$. 
Degree-k Spanning Tree (k-ST)

**Problem**: Given that Hamiltonian-Path is NP-complete, prove that degree-k spanning tree is NP-complete.

**Step 1**: $k$-ST $\in$ NP

A spanning tree acts as a certificate. The certifier checks that $T$ is a spanning tree of $G$ and that each vertex has degree less than $k$.

**Step 2**: HAMILTONIAN-PATH $\leq_p$ k-ST

**Goal**: Suppose that we have an efficient algorithm for solving Degree-k spanning tree, it can simply be used to decide whether $G$ has a Hamiltonian path.
Degree-k Spanning Tree (k-ST)

**Step 2:** $\text{HP} \leq_p k\text{-ST}$

$G$ has a Hamiltonian path if and only if it has a spanning tree with vertex degree $\leq 2$ ($k = 2$).

- **If part** - It is easy to see that such a spanning tree is a Hamiltonian path. Since all vertices in the tree have degree $\leq 2$ it cannot branch and since it is spanning only two vertices can have degree $< 2$. So the spanning tree is a Hamiltonian path.

- **Only if part** - If, on the other hand, $G$ contains a Hamiltonian path, this path must be a spanning tree since the path visits every node and a path trivially is a tree.
Problem: Given that Hamiltonian-Path is NP-complete, prove that degree-k spanning tree is NP-complete.

Step 2: HP \leq_p k-ST

\( G \) has a Hamiltonian path if and only if it has a spanning tree with vertex degree \( \leq 2 \) (\( k = 2 \)).

- **If part** - It is easy to see that such a spanning tree is a Hamiltonian path. Since all vertices in the tree have degree \( \leq 2 \) it cannot branch and since it is spanning only two vertices can have degree \( < 2 \). So the spanning tree is a Hamiltonian path.

- **Only if part** - If, on the other hand, \( G \) contains a Hamiltonian path, this path must be a spanning tree since the path visits every node and a path trivially is a tree.
Degree-k Spanning Tree (k-ST)

Problem: Given that Hamiltonian-Path is NP-complete, prove that degree-k spanning tree is NP-complete.

Step 2: HP $\leq_p$ k-ST

Suppose that we have an efficient algorithm for solving k-ST, it can simply be used to decide whether $G$ has a Hamiltonian path by asking it to determine whether $(G, 2)$ is a yes-instance of k-ST.
**Problem**: Given that Vertex Cover is NP-complete, prove that VC-EVEN is NP-complete.

**VC-EVEN**: Given an undirected graph $G$ in which every vertex has even degree, and an integer $k$, does $G$ have a vertex cover with $k$ vertices?
Problem: Given that Vertex Cover is NP-complete, prove that VC-EVEN is NP-complete.

Step 1: VC-EVEN ∈ NP
VC-EVEN is a special case of VC it follows trivially that VC-EVEN ∈ NP

Step 2: VC ≤ₚ VC-EVEN
Consider the graph $G'$ constructed from graph $G$ such that every vertex in $G'$ has even degree.

Goal: Suppose that we have an efficient algorithm for solving VC-EVEN given $(G', k')$, it can simply be used to decide whether $G$ has a vertex cover of size $k$, by asking it to determine whether $(G', k')$ is a yes-instance of VC-EVEN.
Even VC

**Problem**: Given that Vertex Cover is NP-complete, prove that VC-EVEN is NP-complete.

**Step 2**: $\text{VC} \leq_p \text{VC-EVEN}$

**Proof**: Consider the graph $G'$ constructed from graph $G$ such that every vertex in $G'$ has even degree.

$G' = (V', E')$ will be constructed as follows: if $G = (V, E)$, then $V' = V \cup \{v_1, v_2, v_3\}$, and $E' = E \cup \{(v_1, v_2), (v_2, v_3), (v_1, v_3)\} \cup \{(v, v_1) | v \in V \text{ and } v \text{ has odd degree}\}$. And, finally, $k = k' + 2$. 
**Problem**: Given that Vertex Cover is NP-complete, prove that VC-EVEN is NP-complete.

**Step 2**: $\text{VC} \leq_p \text{VC-EVEN}$

**Proof**: Consider the graph $G'$ constructed from graph $G$ such that every vertex in $G'$ has even degree.

$G' = (V', E')$ will be constructed as follows: if $G = (V, E)$, then $V' = V \cup \{v_1, v_2, v_3\}$, and $E' = E \cup \{(v_1, v_2), (v_2, v_3), (v_1, v_3)\} \cup \{(v, v_1) | v \in V$ and $v$ has odd degree} And, finally, $k = k' + 2$.

Every vertex in $G'$ has even degree.

- For any graph $G = (V, E)$, the number of odd-degree vertices is even.  
  **Proof**: The sum of the degree of all vertices in a graph with $m$ edges is $2m$ since each edge contributes 1 to the degree of exactly two vertices
Even VC

**Problem**: Given that Vertex Cover is NP-complete, prove that VC-EVEN is NP-complete.

**Step 2**: $\text{VC} \leq_p \text{VC-EVEN}$

**Proof**: $(G, k)$ is a yes-instance of VC if and only if $(G', k')$ is a yes-instance of VC-EVEN.

- **If part** – Assume that $G'$ has a vertex cover of cardinality $k'$. At least two of the vertices $v1, v2, v3$ must be in the vertex cover of $G'$ for that triangle to be covered, therefore, since all edges in $G'$ are covered by $k + 2$ vertices, the remaining edges must be covered by $k$ vertices. These remaining edges are exactly those in $G$. We must also show that every vertex in $G'$ has even degree.

- **Only if part** - Clearly if there is a vertex cover for $G$, $C \subseteq V$ such that $|C| = k$ then $C \cup v1, v3$ is a vertex cover for $G'$ of size $k' = k + 2$. 
Bin Packing

- Given an infinite supply of unit-capacity bins, and a list of items $i_1, i_2, \ldots$, pack the items into a minimum number of bins without exceeding any bin capacity.

- **Online Bin Packing:** Must assign each item to a bin as soon as it arrives (!)
  - Bagging groceries, cutting stock (pipes, lumber), etc.

- **Performance Ratio:** For a given instance (list of items) $L$, the *performance ratio* of an online bin packing heuristic $H$ is the limit, as the number of items in $L \to \infty$, of the maximum ratio
  
  $$(\text{#bins used by } H \text{ to pack } L) / (\text{Opt #bins needed to pack } L)$$
Questions/ Topic to Discuss?