NP Hardness Reductions
Reminders/Check-in

- HW Clarifications
  - We ask you to give a polynomial-time algorithm, so want to justify that your algorithm is polynomial time
    - Rudimentary analysis is OK! But remember that cost of algorithm is cost of reduction + cost of solving/interpreting flow
  - Ford Fulkerson: \(O(nmC)\) or \(O(mC)\)?
    - Textbook uses different definition of \(C\) than we did in our discussion…
- Probability review
  - Readings accessible from on-campus only (or using proxy)
Big Picture

• “Does P = NP?” is an important question in CS
  • Knowing the answer would be nice, but the debate around the question informs our thinking about “hard” problems

• So why are we covering it? What should your takeaways be?
  • Be able to give an operation definition of and describe the P and the NP-complete problem classes
  • Be able to complete and prove a problem reduction beyond the examples we cover together (i.e., apply the reduction framework)
  • Be familiar with a handful of the “classic” NP-hard problems
    • If you hear “Vertex Cover” at a party…
VERTEX-COVER $\leq_p$ SET-COVER
Vertex-Cover

Given a graph $G = (V, E)$, a vertex cover is a subset of vertices $T \subseteq V$ such that for every edge $e = (u, v) \in E$, either $u \in T$ or $v \in T$.

- **VERTEX-COVER decision Problem.** Given a graph $G = (V, E)$ and an integer $k$, does $G$ have a vertex cover of size at most $k$?
Set Cover

Set-Cover. Given a set $U$ of elements, a collection $\mathcal{S}$ of subsets of $U$ and an integer $k$, is there some collection of at most $k$ subsets $S_1, \ldots, S_k$ whose union covers $U$, that is, $U \subseteq \bigcup_{i=1}^{k} S_i$

\[
U = \{ 1, 2, 3, 4, 5, 6, 7 \} \\
S_a = \{ 3, 7 \} \quad \quad \quad S_b = \{ 2, 4 \} \\
S_c = \{ 3, 4, 5, 6 \} \quad \quad \quad S_d = \{ 5 \} \\
S_e = \{ 1 \} \quad \quad \quad S_f = \{ 1, 2, 6, 7 \} \\
k = 2
\]

a set cover instance
Set Cover

Set-Cover. Given a set $U$ of elements, a collection $\mathcal{S}$ of subsets of $U$ and an integer $k$, is there some collection of at most $k$ subsets $S_1, \ldots, S_k$ whose union covers $U$, that is, $U \subseteq \bigcup_{i=1}^{k} S_i$

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$  
$S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$  
$S_d = \{ 5 \}$

$S_e = \{ 1 \}$  
$S_f = \{ 1, 2, 6, 7 \}$

$k = 2$

a set cover instance
**Vertex Cover \( \leq_p \) Set Cover**

- **Theorem.** VERTEX-COVER \( \leq_p \) SET-COVER

- **Proof.** Given instance \( \langle G, k \rangle \) of vertex cover, construct an instance \( \langle U, \mathcal{S}, k' \rangle \) of set cover problem such that

- \( G \) has a vertex cover of size at most \( k \) if and only if \( \langle U, \mathcal{S}, k' \rangle \) has a set cover of size at most \( k \).

![Diagram showing the relationship between Vertex Cover and Set Cover problems](image-url)
**Vertex Cover \( \leq_p \) Set Cover**

- **Theorem.** \( \text{VERTEX-COVER} \leq_p \text{SET-COVER} \)

- **Proof.** Given instance \( \langle G, k \rangle \) of vertex cover, construct an instance \( \langle U, S, k \rangle \) of set cover problem that has a set cover of size \( k \) iff \( G \) has a vertex cover of size \( k \).

- **Reduction.** \( U = E \). \( S \): for each node \( v \in V \), let \( S_v = \{ e \in E \mid e \text{ incident to } v \} \)

\[\begin{array}{c}
\text{vertex cover instance} \\
(k = 2)
\end{array}\]

\[\begin{array}{c}
\text{set cover instance} \\
(k = 2)
\end{array}\]

\[U = \{ e_1, e_2, \ldots, e_7 \} \]
\[S_a = \{ e_3, e_7 \} \quad S_b = \{ e_2, e_4 \} \]
\[S_c = \{ e_3, e_4, e_5, e_6 \} \quad S_d = \{ e_5 \} \]
\[S_e = \{ e_1 \} \quad S_f = \{ e_1, e_2, e_6, e_7 \} \]
Correctness

• **Claim.** \( (\Rightarrow) \) If \( G \) has a vertex cover of size at most \( k \), then \( U \) can be covered using at most \( k \) subsets.

• **Proof.** Let \( X \subseteq V \) be a vertex cover in \( G \)
  
  • Then, \( Y = \{ S_v \mid v \in X \} \) is a set cover of \( U \) of the same size

\[
U = \{ e_1, e_2, \ldots, e_7 \} \\
S_a = \{ e_3, e_7 \} \quad S_b = \{ e_2, e_4 \} \\
S_c = \{ e_3, e_4, e_5, e_6 \} \quad S_d = \{ e_5 \} \\
S_e = \{ e_1 \} \quad S_f = \{ e_1, e_2, e_6, e_7 \}
\]

vertex cover instance 
\((k = 2)\) 

set cover instance 
\((k = 2)\)
Correctness

- **Claim.** (\(\iff\)) If \(U\) can be covered using at most \(k\) subsets then \(G\) has a vertex cover of size at most \(k\).

- **Proof.** Let \(Y \subseteq \mathcal{S}\) be a set cover of size \(k\)
  - Then, \(X = \{v \mid S_v \in Y\}\) is a vertex cover of size \(k\)

![Vertex Cover Instance](image1)

![Set Cover Instance](image2)

\[
U = \{ e_1, e_2, \ldots, e_7 \}
\]

\[
S_a = \{ e_3, e_7 \} \\
S_b = \{ e_2, e_4 \} \\
S_c = \{ e_3, e_4, e_5, e_6 \} \\
S_d = \{ e_5 \} \\
S_e = \{ e_1 \} \\
S_f = \{ e_1, e_2, e_6, e_7 \}
\]
Class Exercise

IND-SET \leq_p \text{ Clique}
Clique

- A **clique** in an undirected graph is a subset of nodes such that every two nodes are connected by an edge. A $k$-clique is a clique that contains $k$ nodes.

- **CLIQUE.** Given a graph $G$ and a number $k$, does $G$ contain a $k$-clique?
Clique

• A **clique** in an undirected graph is a subset of nodes such that every two nodes are connected by an edge. A $k$-clique is a clique that contains $k$ nodes.

• **CLIQUE.** Given a graph $G$ and a number $k$, does $G$ contain a $k$-clique?

• **CLIQUE $\in$ NP**
  
  • Certificate: a subset of vertices
  
  • Poly-time verifier: check is each pair of vertices have an edge between them and if size of subset is $k$
IND-SET to CLIQUE

- **Theorem.** IND-SET $\leq_p$ CLIQUE.

- **In class exercise.** Reduce IND-SET to Clique. Given instance $\langle G, k \rangle$ of independent set, construct an instance $\langle G', k' \rangle$ of clique such that
  - $G$ has independent set of size $k$ iff $G'$ has clique of size $k'$.

![Algorithm for IND-SET](algorithm-diagram.png)

![Algorithm for CLIQUE](algorithm-diagram.png)
Recall: IND-SET

Given a graph $G = (V, E)$, an independent set is a subset of vertices $S \subseteq V$ such that no two of them are adjacent, that is, for any $x, y \in S$, $(x, y) \notin E$

- **IND-SET decision Problem.** Given a graph $G = (V, E)$ and an integer $k$, does $G$ have an independent set of size at least $k$?
**IND-SET to CLIQUE**

- **Theorem.** $\text{IND-SET} \leq_p \text{CLIQUE}$.

- **In class exercise.** Reduce IND-SET to Clique. Given instance $\langle G, k \rangle$ of independent set, construct an instance $\langle G', k' \rangle$ of clique such that
  - $G$ has independent set of size $k$ iff $G'$ has clique of size $k'$.

![Diagram showing the reduction from IND-SET to CLIQUE](image)
Theorem. \( \text{IND-SET} \leq_p \text{CLIQUE} \).

Proof. Given instance \( \langle G, k \rangle \) of independent set, we construct an instance \( \langle G', k' \rangle \) of clique such that \( G \) has independent set of size \( k \) iff \( G' \) has clique of size \( k' \).

Reduction.

- Let \( G' = (V, \overline{E}) \), where \( e = (u, v) \in \overline{E} \) iff \( e \not\in E \) and \( k' = k \).
- \( (\Rightarrow) \) \( G \) has an independent set \( S \) of size \( k \), then \( S \) is a clique in \( G' \).
- \( (\Leftarrow) \) \( G' \) has a clique \( Q \) of size \( k \), then \( Q \) is an independent set in \( G \).
Reductions: General Pattern

• Describe a polynomial-time algorithm to transform an arbitrary instance $x$ of Problem $X$ into a special instance $y$ of Problem $Y$

• Prove that:
  • If $x$ is a “yes” instance of $X$, then $y$ is a “yes” instance of $Y$
  • If $y$ is a “yes” instance of $Y$, then $x$ is a “yes” instance of $X$
Reductions: General Pattern

- Describe a polynomial-time algorithm to transform an arbitrary instance $x$ of Problem $X$ into a special instance $y$ of Problem $Y$.

- Notice that correctness of reductions are not symmetric:
  - the “if” proof needs to handle arbitrary instances of $X$.
  - the “only if” needs to handle the special instance of $Y$.

![Diagram showing the general pattern of reductions between $X$ and $Y$.]
IND-SET is NP Complete:

$$3\text{SAT} \leq_p \text{IND-SET}$$
Problem Definition: 3-SAT

- **Literal.** A Boolean variable or its negation: $x_i$ or $\overline{x_i}$

- **Clause.** A disjunction of literals: $C_j = x_1 \lor \overline{x_2} \lor x_3$

- **Conjunctive normal form (CNF).** A boolean formula $\phi$ that is a conjunction of clauses: $\Phi = C_1 \land C_2 \land C_3$

- **SAT.** Given a CNF formula $\Phi$, does it have a satisfying truth assignment?

- **3SAT.** A SAT formula where each clause contains exactly 3 literals (corresponding to different variables)

  $\Phi = (\overline{x_1} \lor x_2 \lor x_3) \land (x_1 \lor \overline{x_2} \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_4)$

- **SAT, 3SAT** are both NP complete

- We will use 3SAT to prove other problems are NP hard
IND-SET: NP Complete

• To show Independent set is NP complete
  • Show it is in NP (we’ve already done this)
  • Reduce a known NP complete problem to it
    • We will use 3-SAT
  • Looking ahead: once we have shown 3-SAT $\leq_p$ IND-SET
    • Since IND-SET $\leq_p$ Vertex Cover
    • And Vertex Cover $\leq_p$ Set Cover
    • We can conclude they are also NP hard
    • As they are both in NP, they are also NP complete!
**IND-SET: NP hard**

- **Theorem.** $3$-SAT $\leq_p$ IND-SET

- Given an instance $\Phi$ of 3-SAT, we construct an instance $\langle G, k \rangle$ of IND-SET s.t. $G$ has an independent set of size $k$ iff $\phi$ is satisfiable.
Map the Problems

3SAT

What is a possible solution?

Ind-Set

A selection of vertices to be an IS $S$

What is the requirement?

Each clause must contain at least one literal that is True

$S$ must contain at least $k$ vertices

What are the restrictions?

$x$ can be true iff $\overline{x}$ is assigned false

If $(u, v) \in E$, then both $u$ and $v$ cannot be in $S$
**3SAT** \(\leq_p\) **IND-SET**

- **Reduction.** Let \(k\) be the number of clauses in \(\Phi\).
  - \(G\) has \(3k\) vertices, one for each literal in \(\Phi\)
  - *(Clause gadget)* For each clause, connect the three literals in a triangle
  - *(Variable gadget)* Each variable is connected to its negation

\[\Phi = (\overline{x}_1 \lor x_2 \lor x_3) \land (x_1 \lor \overline{x}_2 \lor x_3) \land (\overline{x}_1 \lor x_2 \lor x_4)\]
3SAT \leq_p IND-SET

- Observations.
  - Any independent set in $G$ can contain at most 1 vertex from each clause triangle
  - Only one of $x_i$ or $\overline{x_i}$ can be in an independent set (consistency)

\[ \Phi = (\overline{x_1} \lor x_2 \lor x_3) \land (x_1 \lor \overline{x_2} \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_4) \]
**3SAT** \( \leq_p \) **IND-SET**

- **Claim.** \( \Phi \) is satisfiable iff \( G \) has an independent set of size \( k \)

- \( (\Rightarrow) \) Suppose \( \Phi \) is satisfiable, consider a satisfying assignment
  - There is at least one true literal in each clause
  - Select one true literal from each clause/triangle
  - This is an independent set of size \( k \)

\[
\Phi = (\bar{x}_1 \lor x_2 \lor x_3) \land (x_1 \lor \bar{x}_2 \lor x_3) \land (\bar{x}_1 \lor x_2 \lor x_4)
\]
3SAT $\leq_p$ IND-SET

- **Claim.** $\Phi$ is satisfiable iff $G$ has an independent set of size $k$
- $(\Leftarrow)$ Let $S$ be in an independent set in $G$ of size $k$
  - $S$ must contain exactly one node in each triangle
  - Set the corresponding literals to *true*
  - Set remaining literals consistently
- All clauses are satisfied — $\Phi$ is satisfiable $\blacksquare$

$$\Phi = (\overline{x_1} \lor x_2 \lor x_3) \land (x_1 \lor \overline{x_2} \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_4)$$
3SAT $\leq_p$ IND-SET

- Our reduction is clearly polynomial time in the input
  - $G$ has $3k$ nodes, where $k$ is #clauses, and $n$ edges (one for each variable in $G$)
- Since independent set is in NP (shown previously)
  - Independent set is NP complete

$\Phi = (\overline{x_1} \lor x_2 \lor x_3) \land (x_1 \lor \overline{x_2} \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_4)$
Reduction Strategies

• Equivalence
  • $\text{VERTEX-COVER} \equiv_p \text{IND-SET}$

• Special case to general case
  • $\text{VERTEX-COVER} \leq_p \text{SET-COVER}$

• Encoding with gadgets
  • $\text{3-SAT} \leq_p \text{IND-SET}$

• Transitivity
  • $\text{3-SAT} \leq_p \text{IND-SET} \leq_p \text{VERTEX-COVER} \leq_p \text{SET-COVER}$

• Thus, $\text{IND-SET}$, $\text{VERTEX-COVER}$ and $\text{SET-COVER}$ are NP hard

• Since they are all in NP, also NP-complete
List of NPC Problems So Far

• 3-SAT
• INDEPENDENT SET
• VERTEX COVER
• SET COVER
• CLIQUE
• More to come:
  • Subset Sum
  • Knapsack
  • 3-COLOR
  • Hamiltonian cycle / path
  • TSP
Steps to Prove $X$ is NP Complete

- Step 1. Show $X$ is in NP
- Step 2. Pick a known NP hard problem $Y$ from class
- Step 3. Show that $Y \leq_p X$
  - Show both sides of reduction are correct: if and only if directions
  - State that reduction runs in polynomial time in input size of problem $Y$
Acknowledgments

• Some of the material in these slides are taken from


  • Jeff Erickson’s Algorithms Book (http://jeffe.cs.illinois.edu/teaching/algorithms/book/Algorithms-JeffE.pdf)