NP-Completeness Proofs

Algorithm Design & Analysis

Spring 2018
NP-Completeness Recap

A decision problem $X$ is NP-Complete if

- $X \in \text{NP}$
- For every $Y \in \text{NP}$, $Y \leq_p X$

Theorem: Let $Y$ be any NP-Complete problem. Then $Y \in \text{P}$ if and only if $\text{P} = \text{NP}$

There are two ways to show a problem $Y$ is NP-Complete

Definition

- Show that $Y \in \text{NP}$
- Show that for all $X \in \text{NP}$, $X \leq_p Y$

Reduction

- Show that $Y \in \text{NP}$
- Show that $Z \leq_p Y$ for some for some NP-Complete problem $Z$
NP-Completeness Recap

A decision problem $X$ is *NP-Complete* if

- $X \in NP$
- For every $Y \in NP$, $Y \leq_p X$

Theorem:
Let $Y$ be any NP-Complete problem. Then $Y \in P$ if and only if $P = NP$

There are two ways to show a problem $Y$ is NP-Complete

**Definition**
- Show that $Y \in NP$
- Show that for all $X \in NP$, $X \leq_p Y$

**Reduction**
- Show that $Y \in NP$
- Show that $Z \leq_p Y$ for some for some NP-Complete problem $Z$
A decision problem $X$ is \textit{NP-Complete} if
\begin{itemize}
    \item $X \in NP$
    \item For every $Y \in NP$, $Y \leq_p X$
\end{itemize}

\textbf{Theorem:} Let $Y$ be \textit{any} NP-Complete problem. Then $Y \in P$ if and only if $P = NP$
NP-Completeness Recap

- A decision problem $X$ is $NP$-Complete if
  - $X \in NP$
  - For every $Y \in NP$, $Y \leq_p X$

- **Theorem:** Let $Y$ be any $NP$-Complete problem. Then $Y \in P$ if and only if $P = NP$

- There are two ways to show a problem $Y$ is $NP$-Complete
  
  **Definition**
  - Show that $Y \in NP$
  - Show that for all $X \in NP$, $X \leq_p Y$
NP-Completeness Recap

- A decision problem $X$ is \textit{NP-Complete} if
  - $X \in NP$
  - For every $Y \in NP$, $Y \leq_p X$

- \textbf{Theorem:} Let $Y$ be any NP-Complete problem. Then $Y \in P$ if and only if $P = NP$

- There are two ways to show a problem $Y$ is NP-Complete
  - \textit{Definition}:
    - Show that $Y \in NP$
    - Show that for all $X \in NP$, $X \leq_p Y$
NP-Completeness Recap

• A decision problem $X$ is $NP$-Complete if
  • $X \in NP$
  • For every $Y \in NP$, $Y \leq_p X$

• **Theorem**: Let $Y$ be any $NP$-Complete problem. Then $Y \in P$ if and only if $P = NP$

• There are two ways to show a problem $Y$ is $NP$-Complete

  **Definition**
  • Show that $Y \in NP$
  • Show that for all $X \in NP$, $X \leq_p Y$

  **Reduction**
  • Show that $Y \in NP$
A decision problem $X$ is \textit{NP-Complete} if
- $X \in \text{NP}$
- For every $Y \in \text{NP}$, $Y \leq_p X$

\textbf{Theorem:} Let $Y$ be any NP-Complete problem. Then $Y \in P$ if and only if $P = \text{NP}$

There are two ways to show a problem $Y$ is NP-Complete

\textit{Definition} 
- Show that $Y \in \text{NP}$
- Show that for all $X \in \text{NP}$, $X \leq_p Y$

\textit{Reduction} 
- Show that $Y \in \text{NP}$
- Show that $Z \leq_p Y$ for some NP-Complete problem $Z$
**Figure 8.4** A circuit with three inputs, two additional sources that have assigned truth values, and one output.
Boolean Circuits

A boolean circuit is a DAG in which

Theorem: Let $A$ be a poly-time algorithm that takes $n$ input bits and produces 1 output bit. Then there is a boolean circuit $C$ that can be produced from $A$ in poly-time such that $C$ produces a 1 if and only if $A$ does.

CIRCUITSAT: Given a boolean circuit $C$ with $n$ input bits (some of which may be fixed), is there an assignment of values to the unfixed input bits such that $C$ returns 1 (true/yes)?

Theorem: CIRCUITSAT is NP-complete
Boolean Circuits

A *boolean circuit* is a DAG in which

- Sources represent input bits

Theorem:
Let $A$ be a poly-time algorithm that takes $n$ input bits and produces 1 output bit. Then there is a boolean circuit $C$ that can be produced from $A$ in poly-time such that $C$ produces a 1 if and only if $A$ does.

CIRCUIT SAT:
Given a boolean circuit $C$ with $n$ input bits (some of which may be fixed), is there an assignment of values to the unfixed input bits such that $C$ returns 1 (true/yes)?

Theorem:
CIRCUIT SAT is NP-complete
Boolean Circuits

A boolean circuit is a DAG in which
- Sources represent input bits
- Sinks represent output bits
Boolean Circuits

A boolean circuit is a DAG in which

- Sources represent input bits
- Sinks represent output bits
- Other bits represent boolean operations ($\land$, $\lor$, $\neg$)

Theorem: Let $A$ be a poly-time algorithm that takes $n$ input bits and produces 1 output bit. Then there is a boolean circuit $C$ that can be produced from $A$ in poly-time such that $C$ produces a 1 if and only if $A$ does.

CIRCUITSAT: Given a boolean circuit $C$ with $n$ input bits (some of which may be fixed), is there an assignment of values to the unfixed input bits such that $C$ returns 1 (true/yes)?

Theorem: CIRCUITSAT is NP-complete
A boolean circuit is a DAG in which

- Sources represent input bits
- Sinks represent output bits
- Other bits represent boolean operations ($\land$, $\lor$, $\neg$)

**Theorem:** Let $A$ be a poly-time algorithm that takes $n$ input bits and produces 1 output bit. Then there is a boolean circuit $C$ that can be produced from $A$ in poly-time such that $C$ produces a 1 if and only if $A$ does
Boolean Circuits

A boolean circuit is a DAG in which

- Sources represent input bits
- Sinks represent output bits
- Other bits represent boolean operations ($\land, \lor, \lnot$)

**Theorem:** Let $A$ be a poly-time algorithm that takes $n$ input bits and produces 1 output bit. Then there is a boolean circuit $C$ that can be produced from $A$ in poly-time such that $C$ produces a 1 if and only if $A$ does

**CIRCUITSAT:** Given a boolean circuit $C$ with $n$ input bits (some of which may be fixed), is there an assignment of values to the unfixed input bits such that $C$ returns 1 (true/yes)?
A *boolean circuit* is a DAG in which
- Sources represent input bits
- Sinks represent output bits
- Other bits represent boolean operations ($\land, \lor, \neg$)

**Theorem:** Let $A$ be a poly-time algorithm that takes $n$ input bits and produces 1 output bit. Then there is a boolean circuit $C$ that can be produced from $A$ in poly-time such that $C$ produces a 1 if and only if $A$ does

**CIRCUITSAT:** Given a boolean circuit $C$ with $n$ input bits (some of which may be fixed), is there an assignment of values to the unfixed input bits such that $C$ returns 1 (true/yes)?

**Theorem:** CIRCUITSAT is NP-complete
From CIRCUITSAT to ATMOST3SAT

Definition
Let $\Phi$ be a CNF expression with at most 3 literals per clause. ATMOST3SAT is the problem of deciding whether $\Phi$ is satisfiable.
From CIRCUITSAT to ATMOST3SAT

Definition
Let $\Phi$ be a CNF expression with at most 3 literals per clause. ATMOST3SAT is the problem of deciding whether $\Phi$ is satisfiable.

Theorem
ATMOST3SAT is NP-complete.
From CIRCUITSAT to ATMOST3SAT

Definition
Let $\Phi$ be a CNF expression with at most 3 literals per clause. ATMOST3SAT is the problem of deciding whether $\Phi$ is satisfiable.

Theorem
ATMOST3SAT is NP-complete.

Idea
Note that ATMOST3SAT is in NP. We show that CIRCUITSAT $\leq_p$ ATMOST3SAT.
From CIRCUITSAT to ATMOST3SAT

Definition
Let $\Phi$ be a CNF expression with at most 3 literals per clause. ATMOST3SAT is the problem of deciding whether $\Phi$ is satisfiable.

Theorem
ATMOST3SAT is NP-complete.

Idea
Note that ATMOST3SAT is in NP. We show that CIRCUITSAT $\leq_p$ ATMOST3SAT.

- Let $C$ be a boolean circuit. We’ll build formula $\Phi_C$ such that $C$ is satisfiable if and only if $\Phi_C$ is satisfiable (or empty).
From CIRCUITSAT to ATMOST3SAT

Definition
Let $\Phi$ be a CNF expression with at most 3 literals per clause. ATMOST3SAT is the problem of deciding whether $\Phi$ is satisfiable.

Theorem
ATMOST3SAT is NP-complete.

Idea
Note that ATMOST3SAT is in NP. We show that CIRCUITSAT $\leq_P$ ATMOST3SAT.

- Let $C$ be a boolean circuit. We’ll build formula $\Phi_C$ such that $C$ is satisfiable if and only if $\Phi_C$ is satisfiable (or empty).
- But first we’ll develop some gadgets
Some building blocks
Gadget Design

Some building blocks

- If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:

$$\Phi_C = (x_v \lor \neg x_u) \land (\neg x_v \lor x_u \lor x_w)$$

- If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:

$$\Phi_C = (x_v \lor \neg x_u) \land (x_v \lor \neg x_w) \land (\neg x_v \lor x_u \lor x_w)$$

- If $C$ is an $\land$-gate $v$ with incoming edges $uv$ and $wv$:

$$\Phi_C = (\neg x_v \lor x_u) \land (\neg x_v \lor x_w) \land (x_v \lor \neg x_u \lor \neg x_w)$$

In each case:

- Any set of values for $u$, $v$, and $w$ consistent with the function of gate $C$ yields an assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$.

- An assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$ yields values to $x_u$, $x_v$, $x_w$ that is consistent with the function of gate $C$. 
Gadget Design

Some building blocks

• If $C$ is a $\neg$-gate with incoming edge $uv$:
  \[
  \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u)
  \]

In each case

• Any set of values for $u$, $v$, and $w$ consistent with the function of gate $C$,
  yields an assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$

• An assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$
  yields values to $x_u$, $x_v$, $x_w$ that is consistent with the function of gate $C$. 
Some building blocks

- If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:
  \[
  \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u)
  \]

- If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:

In each case,
- Any set of values for $u$, $v$, and $w$ consistent with the function of gate $C$ yields an assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$.
- An assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$ yields values to $x_u$, $x_v$, $x_w$ that is consistent with the function of gate $C$. 


Gadget Design

Some building blocks

• If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:
  \[ \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u) \]

• If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (x_v \lor \bar{x}_u) \land (x_v \lor \bar{x}_w) \land (\bar{x}_v \lor x_u \lor x_w) \]
Some building blocks

- If \( C \) is a \( \neg \)-gate \( v \) with incoming edge \( uv \):
  \[
  \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u)
  \]

- If \( C \) is an \( \lor \)-gate \( v \) with incoming edges \( uv \) and \( wv \):
  \[
  \Phi_C = (x_v \lor \bar{x}_u) \land (x_v \lor \bar{x}_w) \land (\bar{x}_v \lor x_u \lor x_w)
  \]

- If \( C \) is an \( \land \)-gate \( v \) with incoming edges \( uv \) and \( wv \):
Some building blocks

- If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:
  \[
  \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u)
  \]

- If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:
  \[
  \Phi_C = (x_v \lor \bar{x}_u) \land (x_v \lor \bar{x}_w) \land (\bar{x}_v \lor x_u \lor x_w)
  \]

- If $C$ is an $\land$-gate $v$ with incoming edges $uv$ and $wv$:
  \[
  \Phi_C = (\bar{x}_v \lor x_u) \land (\bar{x}_v \lor x_w) \land (x_v \lor \bar{x}_u \lor \bar{x}_w)
  \]
Some building blocks

- If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:
  \[ \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u) \]

- If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (x_v \lor \bar{x}_u) \land (x_v \lor \bar{x}_w) \land (\bar{x}_v \lor x_u \lor x_w) \]

- If $C$ is an $\land$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (\bar{x}_v \lor x_u) \land (\bar{x}_v \lor x_w) \land (x_v \lor \bar{x}_u \lor \bar{x}_w) \]

In each case...

The formulas above describe the behavior of each type of gate $C$ in terms of the incoming edges and variables $x_u$, $x_v$, and $x_w$. Any set of values for $u$, $v$, and $w$ consistent with the function of gate $C$ yields an assignment of values to $x_u$, $x_v$, and $x_w$ that satisfies $\Phi_C$. Conversely, any assignment of values to $x_u$, $x_v$, and $x_w$ that satisfies $\Phi_C$ yields values to $u$, $v$, and $w$ that are consistent with the function of gate $C$. This allows for the construction of complex circuits from these basic building blocks.
Gadget Design

Some building blocks

• If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:
  \[ \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u) \]

• If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (x_v \lor \bar{x}_u) \land (x_v \lor \bar{x}_w) \land (\bar{x}_v \lor x_u \lor x_w) \]

• If $C$ is an $\land$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (\bar{x}_v \lor x_u) \land (\bar{x}_v \lor x_w) \land (x_v \lor \bar{x}_u \lor \bar{x}_w) \]

In each case

• Any set of values for $u$, $v$, and $w$ consistent with the function of gate $C$, yields an assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$
Gadget Design

Some building blocks

- If $C$ is a $\neg$-gate $v$ with incoming edge $uv$:
  \[ \Phi_C = (x_v \lor x_u) \land (\bar{x}_v \lor \bar{x}_u) \]
- If $C$ is an $\lor$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (x_v \lor \bar{x}_u) \land (x_v \lor \bar{x}_w) \land (\bar{x}_v \lor x_u \lor x_w) \]
- If $C$ is an $\land$-gate $v$ with incoming edges $uv$ and $wv$:
  \[ \Phi_C = (\bar{x}_v \lor x_u) \land (\bar{x}_v \lor x_w) \land (x_v \lor \bar{x}_u \lor \bar{x}_w) \]

In each case

- Any set of values for $u$, $v$, and $w$ consistent with the function of gate $C$, yields an assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$
- An assignment of values to $x_u$, $x_v$, $x_w$ that satisfies $\Phi_C$ yields values to $x_u$, $x_v$, $x_w$ that is consistent with the function of gate $C$. 

We need any satisfying assignment for $\Phi_C$ to ensure that output bit equals 1 and that fixed input bits of $C$ are set properly:

- For fixed input bit $v$ in $C$: if $v$ is set to 1, $\Phi_C = (x_v)$, else $\Phi_C = (\overline{x_v})$.

Let's look at an example....

Claim For any boolean circuit $C$ with 1 output bit, satisfying assignments of $C$ yield satisfying assignments of $\Phi_C$ (each $x_v$ gets value of $v$) and vice-versa.
We need any satisfying assignment for $\Phi_C$ to ensure that output bit equals 1 and that fixed input bits of $C$ are set properly:

- For fixed input bit $v$ in $C$: if $v$ is set to 1, $\Phi_C = (x_v)$, else $\Phi_C = (\overline{x}_v)$. 

Let's look at an example....
Final Gadgets

We need any satisfying assignment for $\Phi_C$ to ensure that output bit equals 1 and that fixed input bits of $C$ are set properly:

- For fixed input bit $v$ in $C$: if $v$ is set to 1, $\Phi_C = (x_v)$, else $\Phi_C = (\overline{x_v})$.
- For output bit $v$ of $C$, $\Phi_C = (x_v)$
Final Gadgets

We need any satisfying assignment for $\Phi_C$ to ensure that output bit equals 1 and that fixed input bits of $C$ are set properly:

- For fixed input bit $v$ in $C$: if $v$ is set to 1, $\Phi_C = (x_v)$, else $\Phi_C = (\overline{x_v})$.
- For output bit $v$ of $C$, $\Phi_C = (x_v)$

Let’s look at an example....
We need any satisfying assignment for $\Phi_C$ to ensure that output bit equals 1 and that fixed input bits of $C$ are set properly:

- For fixed input bit $v$ in $C$: if $v$ is set to 1, $\Phi_C = (x_v)$, else $\Phi_C = (\bar{x}_v)$.
- For output bit $v$ of $C$, $\Phi_C = (x_v)$

Let’s look at an example....

**Claim**

*For any boolean circuit $C$ with 1 output bit, satisfying assignments of $C$ yield satisfying assignments of $\Phi_C$ (each $x_v$ gets value of $v$) and vice-versa.*
Proof that CIRCUITSAT $\leq_p$ ATMOST3SAT

$C$ satisfiable $\Rightarrow \Phi_C$ satisfiable
Proof that \text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}

\[ C \text{ satisfiable} \implies \Phi_C \text{ satisfiable} \]

Proof:

\[ \text{A satisfying assignment to the inputs of } C \text{ yields values for all other nodes of } C \]
\[ \text{The output node gets value } true \]
\[ \text{For each internal node of } \Phi_C, \text{ the set of corresponding clauses are all satisfied (by construction)} \]
\[ \text{Since the output node has value } true, \text{ the single clause of } \Phi_C \text{ corresponding to it does also} \]
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$C$ satisfiable $\Rightarrow \Phi_C$ satisfiable

Proof:
- A satisfying assignment to the inputs of $C$ yields values for all other nodes of $C$
Proof that \( \text{CIRCUITSAT} \leq_p \text{ATMOST3SAT} \)

\( C \) satisfiable \( \Rightarrow \) \( \Phi_C \) satisfiable

Proof:

- A satisfying assignment to the inputs of \( C \) yields values for all other nodes of \( C \)
- The output node gets value true
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$C$ satisfiable $\Rightarrow \Phi_C$ satisfiable

Proof:
- A satisfying assignment to the inputs of $C$ yields values for all other nodes of $C$
- The output node gets value true
- For each internal node of $\Phi_C$, the set of corresponding clauses are all satisfied (by construction)
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$C$ satisfiable $\Rightarrow \Phi_C$ satisfiable

Proof:
- A satisfying assignment to the inputs of $C$ yields values for all other nodes of $C$
- The output node gets value true
- For each internal node of $\Phi_C$, the set of corresponding clauses are all satisfied (by construction)
- Since the output node has value true, the single clause of $\Phi_C$ corresponding to it does also
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$\Phi_C$ satisfiable $\Rightarrow$ $C$ satisfiable
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$\Phi_C$ satisfiable $\Rightarrow$ $C$ satisfiable

- Assume we have a satisfying assignment $S$ for $\Phi_C$. $S$ makes each clause of $\Phi_C$ true
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$\Phi_C$ satisfiable $\Rightarrow$ $C$ satisfiable

- Assume we have a satisfying assignment $S$ for $\Phi_C$. $S$ makes each clause of $\Phi_C$ true
- Assign to each input bit $v$ of $C$ the value of $x_v$ in $S$
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$\Phi_C$ satisfiable $\Rightarrow$ $C$ satisfiable

- Assume we have a satisfying assignment $S$ for $\Phi_C$. $S$ makes each clause of $\Phi_C$ true
- Assign to each input bit $v$ of $C$ the value of $x_v$ in $S$
- This induces values on every other node of $C$. 
Proof that CIRCUITSAT $\leq_p$ ATMOST3SAT

$\Phi_C$ satisfiable $\Rightarrow$ $C$ satisfiable

- Assume we have a satisfying assignment $S$ for $\Phi_C$. $S$ makes each clause of $\Phi_C$ true
- Assign to each input bit $v$ of $C$ the value of $x_v$ in $S$
- This induces values on every other node of $C$.
- By construction of $\Phi_C$ the values induced on any node $v$ is the value of $x_v$ in $S$
Proof that CIRCUITSAT ≤ₚ ATMOST3SAT

\( \Phi_C \) satisfiable \( \Rightarrow \) \( C \) satisfiable

- Assume we have a satisfying assignment \( S \) for \( \Phi_C \). \( S \) makes each clause of \( \Phi_C \) true
- Assign to each input bit \( v \) of \( C \) the value of \( x_v \) in \( S \)
- This induces values on every other node of \( C \).
- By construction of \( \Phi_C \) the values induced on any node \( v \) is the value of \( x_v \) in \( S \)
- In particular, the output bit \( t \) of \( C \) gets value 1, since \((x_t)\) is a clause of \( \Phi_C \)
Proof that $\text{CIRCUITSAT} \leq_p \text{ATMOST3SAT}$

$\Phi_C$ satisfiable $\Rightarrow$ $C$ satisfiable

- Assume we have a satisfying assignment $S$ for $\Phi_C$. $S$ makes each clause of $\Phi_C$ true
- Assign to each input bit $v$ of $C$ the value of $x_v$ in $S$
- This induces values on every other node of $C$
- By construction of $\Phi_C$ the values induced on any node $v$ is the value of $x_v$ in $S$
- In particular, the output bit $t$ of $C$ gets value 1, since $(x_t)$ is a clause of $\Phi_C$
- Thus $C$ is satisfiable
3SAT is NP-Complete

$3SAT \in NP$. Let’s show that $\text{ATMOST3SAT} \leq_p 3SAT$. 

The Gadget:

Let $z_1, z_2, z_3, z_4$ be boolean variables. Let

$\Phi_1 = (\overline{z}_1 \lor z_3 \lor z_4) \land (\overline{z}_1 \lor \overline{z}_3 \lor z_4) \land (\overline{z}_1 \lor z_3 \lor \overline{z}_4) \land (\overline{z}_1 \lor \overline{z}_3 \lor \overline{z}_4)$

$\Phi_2 = (\overline{z}_2 \lor z_3 \lor z_4) \land (\overline{z}_2 \lor \overline{z}_3 \lor z_4) \land (\overline{z}_2 \lor z_3 \lor \overline{z}_4) \land (\overline{z}_2 \lor \overline{z}_3 \lor \overline{z}_4)$

Claim: $\Phi_1 \land \Phi_2$ is satisfiable exactly when $z_1 = z_2 = 0$.

Let $\Phi \in \text{ATMOST3SAT}$, and let $z_1, \ldots, z_4$ be 4 variables NOT occurring in $\Phi$, and let $C$ be a clause of $\Phi$ with at most 2 literals.

- If $C = (l_1 \lor l_2)$, replace $C$ with $C' = (l_1 \lor l_2 \lor z_1)$
- If $C = (l_1)$, replace $C$ with $C' = (l_1 \lor z_1 \lor z_2)$

Now add $\Phi_1 \land \Phi_2$ and call the modified expression $\Phi'$. Claim: $\Phi'$ is satisfiable if and only if $\Phi'$ is satisfiable.
3SAT is NP-Complete

3SAT ∈ NP. Let’s show that ATMOST3SAT ≤_p 3SAT.

The Gadget: Let z_1, z_2, z_3, z_4 be boolean variables. Let
3SAT is NP-Complete

3SAT ∈ NP. Let’s show that ATMOST3SAT ≤p 3SAT.

The Gadget: Let $z_1, z_2, z_3, z_4$ be boolean variables. Let

$$
\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)
$$

$$
\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)
$$
3SAT is NP-Complete

3SAT ∈ \textit{NP}. Let’s show that \textit{ATMOST3SAT} \leq_p 3SAT.

\textbf{The Gadget:} Let \(z_1, z_2, z_3, z_4\) be boolean variables. Let

\[
\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)
\]

\[
\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)
\]

\textbf{Claim:} \(\Phi_1 \land \Phi_2\) is satisfiable exactly when \(z_1 = z_2 = 0\)
3SAT is NP-Complete

3SAT ∈ NP. Let’s show that ATMOST3SAT \leq_p 3SAT.

The Gadget: Let \( z_1, z_2, z_3, z_4 \) be boolean variables. Let

\[
\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)
\]

\[
\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)
\]

Claim: \( \Phi_1 \land \Phi_2 \) is satisfiable exactly when \( z_1 = z_2 = 0 \)

Let \( \Phi \in ATMOST3SAT \), and let \( z_1, \ldots z_4 \) be 4 variables NOT occurring in \( \Phi \), and let \( C \) be a clause of \( \Phi \) with at most 2 literals.
3SAT is NP-Complete

3SAT ∈ NP. Let’s show that ATMOST3SAT ≤ₚ 3SAT.

The Gadget: Let \(z_1, z_2, z_3, z_4\) be boolean variables. Let

\[
\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)
\]

\[
\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)
\]

Claim: \(\Phi_1 \land \Phi_2\) is satisfiable exactly when \(z_1 = z_2 = 0\)

Let \(\Phi \in ATMOST3SAT\), and let \(z_1, \ldots, z_4\) be 4 variables NOT occurring in \(\Phi\), and let \(C\) be a clause of \(\Phi\) with at most 2 literals.

- If \(C = (l_1 \lor l_2)\), replace \(C\) with \(C' = (l_1 \lor l_2 \lor z_1)\)
3SAT is NP-Complete

3SAT ∈ NP. Let’s show that ATMOST3SAT ≤p 3SAT.

The Gadget: Let $z_1, z_2, z_3, z_4$ be boolean variables. Let

$\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)$

$\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)$

Claim: $\Phi_1 \land \Phi_2$ is satisfiable exactly when $z_1 = z_2 = 0$

Let $\Phi \in ATMOST3SAT$, and let $z_1, \ldots, z_4$ be 4 variables NOT occurring in $\Phi$, and let $C$ be a clause of $\Phi$ with at most 2 literals.

- If $C = (l_1 \lor l_2)$, replace $C$ with $C' = (l_1 \lor l_2 \lor z_1)$
- If $C = (l_1)$, replace $C$ with $C' = (l_1 \lor z_1 \lor z_2)$
3SAT is NP-Complete

3SAT \in NP. Let’s show that ATMOST3SAT \leq_p 3SAT. 

The Gadget: Let \( z_1, z_2, z_3, z_4 \) be boolean variables. Let 

\[
\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)
\]

\[
\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)
\]

Claim: \( \Phi_1 \land \Phi_2 \) is satisfiable exactly when \( z_1 = z_2 = 0 \)

Let \( \Phi \in ATMOST3SAT \), and let \( z_1, \ldots z_4 \) be 4 variables NOT occurring in \( \Phi \), and let \( C \) be a clause of \( \Phi \) with at most 2 literals.

- If \( C = (l_1 \lor l_2) \), replace \( C \) with \( C' = (l_1 \lor l_2 \lor z_1) \)
- If \( C = (l_1) \), replace \( C \) with \( C' = (l_1 \lor z_1 \lor z_2) \)
- Now add \( \Phi_1 \land \Phi_2 \) and call the modified expression \( \Phi' \).
3SAT is NP-Complete

3SAT $\in$ NP. Let’s show that ATMOST3SAT $\leq_p$ 3SAT.

The Gadget: Let $z_1, z_2, z_3, z_4$ be boolean variables. Let

$\Phi_1 = (\bar{z}_1 \lor z_3 \lor z_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_1 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_1 \lor \bar{z}_3 \lor \bar{z}_4)$

$\Phi_2 = (\bar{z}_2 \lor z_3 \lor z_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor z_4) \land (\bar{z}_2 \lor z_3 \lor \bar{z}_4) \land (\bar{z}_2 \lor \bar{z}_3 \lor \bar{z}_4)$

Claim: $\Phi_1 \land \Phi_2$ is satisfiable exactly when $z_1 = z_2 = 0$

Let $\Phi \in$ ATMOST3SAT, and let $z_1, \ldots, z_4$ be 4 variables NOT occurring in $\Phi$, and let $C$ be a clause of $\Phi$ with at most 2 literals.

- If $C = (l_1 \lor l_2)$, replace $C$ with $C' = (l_1 \lor l_2 \lor z_1)$
- If $C = (l_1)$, replace $C$ with $C' = (l_1 \lor z_1 \lor z_2)$
- Now add $\Phi_1 \land \Phi_2$ and call the modified expression $\Phi'$.

Claim: $\Phi'$ is satisfiable if and only if $\Phi'$ is satisfiable.
We’ve established that all of the following problems are NP-Complete
NP-Complete Problems So Far

We’ve established that all of the following problems are NP-Complete

- CIRCUITSAT : direct proof from definition of NP-Complete
NP-Complete Problems So Far

We’ve established that all of the following problems are NP-Complete

- CIRCUITSAT: direct proof from definition of NP-Complete
- ATMOST3SAT: reduction from CIRCUITSAT
- 3SAT: reduction from ATMOST3SAT
- SAT: (obvious) reduction from 3SAT
- INDSET, VERTEXCOVER, SETCOVER: previous reductions, starting with 3SAT

Let’s continue to expand the list.

To show a new problem $X$ is NP-complete, we can reduce to it from any of our known NP-complete problems.
We’ve established that all of the following problems are NP-Complete:

- CIRCUI TSAT: direct proof from definition of NP-Complete
- AT MOST3SAT: reduction from CIRCUI TSAT
- 3SAT: reduction from AT MOST3SAT
- SAT: (obvious) reduction from 3SAT
- INDSET, VERTEXCOVER, SETCOVER: previous reductions, starting with 3SAT

Let’s continue to expand the list. To show a new problem \( X \) is NP-complete, we can reduce \( X \) to any of our known NP-complete problems.
NP-Complete Problems So Far

We’ve established that all of the following problems are NP-Complete

- CIRCUITSAT: direct proof from definition of NP-Complete
- ATMOST3SAT: reduction from CIRCUITSAT
- 3SAT: reduction from ATMOST3SAT
- SAT: (obvious) reduction from 3SAT

Let’s continue to expand the list.

To show a new problem $X$ is NP-complete, we can reduce to it from any of our known NP-complete problems.
NP-Complete Problems So Far

We’ve established that all of the following problems are NP-Complete

- CIRCUITSAT : direct proof from definition of NP-Complete
- ATMOST3SAT : reduction from CIRCUITSAT
- 3SAT : reduction from ATMOST3SAT
- SAT : (obvious) reduction from 3SAT
- INDSET, VERTEXCOVER, SETCOVER : previous reductions, starting with 3SAT
We’ve established that all of the following problems are NP-Complete

- CIRCUITSAT: direct proof from definition of NP-Complete
- ATMOST3SAT: reduction from CIRCUITSAT
- 3SAT: reduction from ATMOST3SAT
- SAT: (obvious) reduction from 3SAT
- INDSET, VERTEXCOVER, SETCOVER: previous reductions, starting with 3SAT

Let’s continue to expand the list.
NP-Complete Problems So Far

We’ve established that all of the following problems are NP-Complete

- CIRCUITSAT : direct proof from definition of NP-Complete
- ATMOST3SAT : reduction from CIRCUITSAT
- 3SAT : reduction from ATMOST3SAT
- SAT : (obvious) reduction from 3SAT
- INDSET, VERTEXCOVER, SETCOVER : previous reductions, starting with 3SAT

Let’s continue to expand the list.

To show a new problem X is NP-complete, we can reduce to it from any of our known NP-complete problems.
SETPACKING is NP-Complete

The Problem: Given a collection $S = \{S_1, \ldots S_n\}$ of subsets of a set $U$ and an integer $k$, is there a collection $S_i_1, \ldots, S_i_k$ of $S$ such that these sets are pairwise disjoint?
**SETPACKING is NP-Complete**

The Problem: Given a collection $\mathcal{S} = \{S_1, \ldots, S_n\}$ of subsets of a set $U$ and an integer $k$, is there a collection $S_{i_1}, \ldots, S_{i_k}$ of $\mathcal{S}$ such that these sets are pairwise disjoint?

Clearly SETPACKING $\in$ NP : certificate is list of $k$ such subsets
SETPACKING is NP-Complete

The Problem: Given a collection \( S = \{S_1, \ldots, S_n\} \) of subsets of a set \( U \) and an integer \( k \), is there a collection \( S_{i_1}, \ldots, S_{i_k} \) of \( S \) such that these sets are pairwise disjoint?

Clearly SETPACKING \( \in \) NP : certificate is list of \( k \) such subsets

Claim: INDSET \( \leq_p \) SETPACKING : Reduction FROM INDSET!
**SETPACKING is NP-Complete**

**The Problem:** Given a collection $\mathcal{S} = \{S_1, \ldots, S_n\}$ of subsets of a set $U$ and an integer $k$, is there a collection $S_{i_1}, \ldots, S_{i_k}$ of $\mathcal{S}$ such that these sets are pairwise disjoint?

Clearly SETPACKING $\in$ NP : certificate is list of $k$ such subsets

**Claim:** INDSET $\leq^p$ SETPACKING : Reduction FROM INDSET!

- Let $G = (V, E)$ be a graph and let $v \in V$. Define $E_v = \{e \in E : e = \{v, u\}$ for some $u \in V\}$
**SETPACKING is NP-Complete**

The Problem: Given a collection $\mathcal{S} = \{S_1, \ldots, S_n\}$ of subsets of a set $U$ and an integer $k$, is there a collection $S_{i_1}, \ldots, S_{i_k}$ of $\mathcal{S}$ such that these sets are pairwise disjoint?

Clearly SETPACKING $\in$ NP: certificate is list of $k$ such subsets

**Claim:** INDSET $\leq^p$ SETPACKING: Reduction FROM INDSET!

- Let $G = (V, E)$ be a graph and let $v \in V$. Define $E_v = \{e \in E : e = \{v, u\} \text{ for some } u \in V\}$
- Note: $X \subseteq V$ is an independent set if and only if $E_u \cap E_v = \emptyset$ for all $u, v \in X$
**SETPACKING is NP-Complete**

**The Problem:** Given a collection $S = \{S_1, \ldots, S_n\}$ of subsets of a set $U$ and an integer $k$, is there a collection $S_{i_1}, \ldots, S_{i_k}$ of $S$ such that these sets are pairwise disjoint?

Clearly SETPACKING $\in$ NP : certificate is list of $k$ such subsets

**Claim:** INDSET $\leq_p$ SETPACKING : Reduction FROM INDSET!

- Let $G = (V, E)$ be a graph and let $v \in V$. Define $E_v = \{e \in E : e = \{v, u\}$ for some $u \in V\}$

- Note: $X \subseteq V$ is an independent set if and only if $E_u \cap E_v = \emptyset$ for all $u, v \in X$

- Given an instance $(G, k)$ of INDSET, create the set $S_G = \{E_v : v \in V\}$
The Problem: Given a collection $S = \{S_1, \ldots, S_n\}$ of subsets of a set $U$ and an integer $k$, is there a collection $S_{i_1}, \ldots, S_{i_k}$ of $S$ such that these sets are pairwise disjoint?

Clearly SETPACKING $\in$ NP : certificate is list of $k$ such subsets

Claim: INDSET $\leq_p$ SETPACKING : Reduction FROM INDSET!

- Let $G = (V, E)$ be a graph and let $v \in V$. Define $E_v = \{e \in E : e = \{v, u\}$ for some $u \in V\}$
- Note: $X \subseteq V$ is an independent set if and only if $E_u \cap E_v = \emptyset$ for all $u, v \in X$
- Given an instance $(G, k)$ of INDSET, create the set $S_G = \{E_v : v \in V\}$
- SETPACKING returns "yes" if and only if there are sets $E_{v_1}, \ldots, E_{v_k}$ of $S_G$ that are pairwise disjoint
**SETPACKING is NP-Complete**

**The Problem:** Given a collection \( S = \{S_1, \ldots, S_n\} \) of subsets of a set \( U \) and an integer \( k \), is there a collection \( S_{i_1}, \ldots, S_{i_k} \) of \( S \) such that these sets are pairwise disjoint?

Clearly SETPACKING \( \in \text{NP} \) : certificate is list of \( k \) such subsets

**Claim:** \( \text{INDSET} \leq_p \text{SETPACKING} \) : Reduction FROM \( \text{INDSET}! \)

- Let \( G = (V, E) \) be a graph and let \( v \in V \). Define \( E_v = \{e \in E : e = \{v, u\} \text{ for some } u \in V\} \)
- Note: \( X \subseteq V \) is an independent set if and only if \( E_u \cap E_v = \emptyset \) for all \( u, v \in X \)
- Given an instance \((G, k)\) of \( \text{INDSET} \), create the set \( S_G = \{E_v : v \in V\} \)
- \( \text{SETPACKING} \) returns "yes" if and only if there are sets \( E_{v_1}, \ldots, E_{v_k} \) of \( S_G \) that are pairwise disjoint
- That is, if and only if \( \{v_1, \ldots, v_k\} \) is an independent set of \( G \).
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories:

• Packing Problems: INDSET, SETPACKING
• "...at least \(k\)...
• Covering Problems: VERTEXCOVER, SETCOVER
• "...at most \(k\)...
• Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

We'll explore three more categories:

• Partition Problems: Packing meets Covering
• Sequencing Problems
• Numerical Problems
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k$..."

- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k$..."

- Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

- Partition Problems: Packing meets Covering
- Sequencing Problems
- Numerical Problems
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k$..."
- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k$..."
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k$..."
- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k$..."
- Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

We'll explore three more categories

- Partition Problems: Packing meets Covering
- Sequencing Problems
- Numerical Problems
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k$..."
- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k$..."
- Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k$..."
- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k$..."
- Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

We’ll explore three more categories
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- **Packing Problems**: INDSET, SETPACKING
  - "...at least $k$..."
- **Covering Problems**: VERTEXCOVER, SETCOVER
  - "...at most $k$..."
- **Constraint Satisfaction Problems**: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

We’ll explore three more categories

- **Partition Problems**: Packing meets Covering
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k$..."
- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k$..."
- Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

We’ll explore three more categories

- Partition Problems: Packing meets Covering
- Sequencing Problems
A Partial Taxonomy of NP-Complete Problems

The NP-complete problems discussed so far fall into three rough categories

- Packing Problems: INDSET, SETPACKING
  - "...at least $k...""
- Covering Problems: VERTEXCOVER, SETCOVER
  - "...at most $k...""
- Constraint Satisfaction Problems: CIRCUITSAT, ATMOST3SAT, 3SAT, SAT

We’ll explore three more categories

- Partition Problems: Packing meets Covering
- Sequencing Problems
- Numerical Problems