First NPC Problem

CIRCUIT-SAT

Combinational Circuit (Boolean)

Nodes: Logic Gates: NOT, OR, AND,

AND may have more than 2 inputs.

All edges go from left to right.

All $x_i$ are 0/1 variables (Boolean)

$\overline{x_i} = 1 - x_i$  $x_1 \lor x_2$ is 1 if at least one of $x_1, x_2$ is 1

$x_1 \land x_2$ is 1 if (and only if) both = 1.

$\overline{x_i} = 1 - x_i$

Decision: Given a boolean Combinational Circuit, are there 0/1 values to the inputs (here $x_1, x_2, x_3$) such that the output (here $x_{10}$) is 1?

This problem is called Circuit Satisfiability (CIRCUIT-SAT for short)
Boolean Formula Satisfiability (SAT)

Def.: An instance of SAT is a boolean formula \( \phi \) composed of:

1. Boolean Variables (0/1 variables)
2. Boolean Connectives: AND (\( \land \)), OR (\( \lor \)), NOT (\( \neg \)), implication (\( \rightarrow \)), if and only if (\( \leftrightarrow \))
3. Parentheses

We have already explained the first three.

\[ x_1 \rightarrow x_2 : \quad \text{if } x_1 = 1, \quad x_2 = 1. \]

\[ x_1 \leftrightarrow x_2 : \quad x_2 = 1 \quad \text{if } x_1 = 1 \]
\[ \quad x_2 = 0 \quad \text{if } x_1 = 0 \]

[Nothing is said about \( x_2 \) if \( x_1 = 0 \)]

Example \( \phi = \left( (x_1 \rightarrow x_2) \lor \neg (\overline{x_1} \leftrightarrow x_3) \lor x_4 \right) \land \overline{x_2} \)

Def.: A truth assignment for \( \phi \) is a set of values for the variables in \( \phi \) [Note: if \( x_i = 1 \), \( \overline{x_i} = 0 \)] and a satisfying assignment is a truth assignment so that \( \phi = 1 \).
If $\phi$ has a satisfying assignment then we say $\phi$ is satisfiable.

**SAT:** Given a boolean formula $\phi$, is it satisfiable?

---

A proof that $\text{SAT} \in \text{NP}$ using the fact that $\text{CIRCUIT-SAT} \in \text{NP}$ is in CLRS and Lect #14

---

**Conjunctive (Disjunctive) Normal Form (CNF/DNF)**

A literal in a boolean formula is a variable or its negation ($\overline{x_i}$).

A boolean formula is in CNF (DNF) if it is expressed as an AND (OR) of clauses, each of which is the OR (AND) of one or more literals.

$\text{CNF} \rightarrow \text{DNF}$ or $\text{DNF} \rightarrow \text{CNF}$ is done using De Morgan Formula. Hence any boolean formula can be expressed in either form.

[In the book & Lect #14, you will see this]

This is used in show
\textbf{K-CNF}: If each clause in CNF form of $\phi$ has exactly $k$ literals, it is called \textbf{K-CNF}.

$k=3$, is \textbf{3-CNF}

\textbf{SAT problem} in which $\phi$ is 3-CNF is called \textbf{3-SAT}

---

Book, Lect 7/4: Prove \textbf{3-SAT} $\in \textbf{NP}$ \textbf{-C}

\textit{using} \textbf{CIRCUIT-SAT} $\in \textbf{NP}$ \textbf{-C}

---

\textbf{3-SAT $\Rightarrow$ CLIQUE}

\textbf{CLIQUE}: Given an undirected graph $G = [V; E]$ and a positive integer $k$,

$Q$: Is there a \textbf{CLIQUE} in $G$ of size $k$?

\textbf{Def.}: A subset $S \leq V$ of vertices in $G$ is called a \textbf{clique} if, for every pair $i \in S$, $j \in S$, there is an edge in $G$, i.e., $i \in S$, $j \in S \Rightarrow (i, j) \in E$.

Size of a clique is the number of vertices in it.

Now we will do our first proof of $\text{NP}$-C.
1) First we show \textbf{CLIQUE ENP}. 

For this, given \( G = (V; E) \) and pos. int \( k \).

We need a \textbf{Verification Certificate} for each instance where there is a clique of size \( k \) in \( G \) which can be used to verify that the answer is indeed \( \text{yes} \) when this is the case. Moreover, this should not be a \textbf{FALSE Certificate}.

If \( G \) has a clique of size \( k \), [and we know it by knowing a \( S \subseteq V, |S| = k \) which is a clique], we produce \( S \) as the Certificate. This requires a 0/1 vector \( \mathbf{v} \) of size \( |V| \):

\[
\begin{align*}
\text{If } S & \ni v_i \Rightarrow \text{that node } \in S \\
0 & \Rightarrow \ldots \notin S
\end{align*}
\]

Now given \( G \), \( k \) and the Certificate \( S \), the Verification alg. checks that \( S \) is a clique of size \( k \) by:

\[
\begin{align*}
\text{if } & |S| \leq k \\
\text{or } & \text{ if } i \in S, j \in S \Rightarrow (i,j) \in E
\end{align*}
\]

This process will weed out the possibility of \textbf{FALSE Certificate}.
Here CLIQUE \( \in \text{NP} \).

Note: Size of certificate is polynomial in the input \( \subset \text{includes G} \).

2) CLIQUE \( \not\in \text{NPC} \)

This is done by selecting a known NPC problem and showing \( A \leq_p \text{CLIQUE} \).

"A" chosen here is 3-SAT

Showing this involves several parts.

3-SAT \( \leq \text{CLIQUE} \)

Arbitrary instance \( \rightarrow \) Corresponding instance of CLIQUE

\[ \phi = C_1 \land C_2 \land \ldots \land C_k \]

\[ C_j = (l_j^{(1)} \lor l_j^{(2)} \lor l_j^{(3)}) \]

\[ l_j^{(i)} \in \{x_{i1}, \overline{x}_{i1}, x_{i2}, \overline{x}_{i2}, \ldots, x_{in}, \overline{x}_{in}\} \]

Variable \& their negation

Size: \( \Theta(f(n,k); \text{f poly}) \)

\[
\text{Must be a single method that works for all instances of 3-SAT such that "results" are same for both problems.}
\]
given an arbitrary instance of 3-SAT

(which may be a "yes" instance or a "no" instance)

we need to describe the "mechanism" (translation:

an algorithm) to construct a "corresponding"

instance (means "yes" instance in 3-SAT is transformed

into a "yes" instance of CLIQUE and a "no" instance

of 3-SAT is transformed into a "no" instance

of CLIQUE using the same algorithm for

transformation).

This part requires creativity just like

producing an algorithm (good one) for a

new problem requires creativity]

We now describe this in general first and then

with an example.

Given: \(C_1, C_2, \ldots, C_k, \ldots\) of 3-SAT

with \(k\) clauses, and \(n\) variables \((x_1, \ldots, x_n)\)

\& their complements.

Needed: Input instance a CLIQUE

i.e. \(G = [V, E]\), \(pos. int k \leq |V|\).

Construction on next page (Stay Tuned)
There are $3k$ vertices in $G$ (where $k$ is the number of clauses in 3-SAT instance) grouped into $k$ sets of 3 vertices each. Call these vertices $l_j^{(r)} : r=1,2,3$ and $j=1,...,k$.

There is an edge between vertices $l_j^{(r)}$ and $l_j^{(s)}$ if and only if $r \neq s$, and $\overline{l_j^{(r)}} \neq \overline{l_j^{(s)}}$.

Recall: Each $l_j^{(r)} \in \{x_1, \overline{x_1}, x_2, \overline{x_2}, ..., x_n, \overline{x_n}\}$

$\overline{r}=1,2,3$

$\overline{j}=1,...,k$

CLIQUE also needs a positive integer $k$ (we need to check if $G$ has a clique of size $k$).

This value of $k$ for CLIQUE is the same as $k'$, the # of clauses in 3-SAT instance.

Construction (transformation from 3-SAT instance to CLIQUE instance) is now complete. But we need to prove that the constructed instance of CLIQUE is a corresponding instance of 3-SAT instance given. Now we give an example.
3-SAT instance

$$\phi = (x_1 \lor \overline{x}_2 \lor \overline{x}_3) \land (\overline{x}_1 \lor x_2 \lor x_3) \land (x_1 \lor x_2 \lor \overline{x}_3)$$

$$k = 3 ; \quad n = 3 ; \quad \ell_1^{(1)} = x_1 , \quad \ell_1^{(2)} = \overline{x}_2 , \quad \ell_1^{(3)} = \overline{x}_3 \quad \ldots$$

$$C_1 :$$

No edge

$$x_1 \Rightarrow \ell_1^{(11)}$$

$$\overline{x}_2 \Rightarrow \ell_2^{(12)}$$

$$x_3 \Rightarrow \ell_3^{(13)}$$

$$\overline{x}_2 \Rightarrow \ell_2^{(22)}$$

$$\overline{x}_2 \Rightarrow \ell_2^{(23)}$$

$$\overline{x}_3 \Rightarrow \ell_3^{(33)}$$

+ many more edges, ... (too many)

No edges between nodes representing same clause.

Corresponding instance

Suppose 3-SAT is "yes" instance: Each clause is "true"; at least one literal in each clause is "true"; [no two can be contradictory]

Circled ones on the first line: $$x_1 = x_2 = x_3 = 1$$

Nodes corresponding to these literals form a clique of size $$K = 3$$.
Hence

"Yes" in 3-SAT \(\implies\) "Yes" in CLIQUE instance
\[
\text{instance} \quad \underline{\text{constructed}}
\]

Next, we need to show the converse of the above.

For this, we note that "Yes" in CLIQUE instance constructed \(\iff\) There is a clique of size 3 in \(G\). It cannot contain more than one from each "group" since no two from the same group are connected by an edge of \(G\). But there are only \(k = 3\) groups.

\[ \therefore "\text{yes}" \text{ in CLIQUE } \iff \text{ exactly one node from each "group" is in the clique.} \]

And these two literals are not contradicting each other (i.e., \(i \neq \overline{j}\)).

\[ \therefore \text{setting these literals } = 1, \text{ is a solution to 3-SAT that makes } \phi = 1. \]

\[ \therefore "\text{yes}" \text{ in CLIQUE } \implies "\text{yes}" \text{ in 3-SAT} \]

This completes the reduction. Also, note that the size of the graph \(G\) is polynomially bounded in \(\text{size of 3-SAT}\).
It also complete "proof" of
\[
\text{CLIQUE} \\
3\text{-SAT} \leq_p \text{CLIQUE}
\]
And hence prov of CLIQUE \text{NP-C}

Next we take up: VERTEX-COVER

**Problem Statement**

*Def*: Given an undir. graph, \( G = [V; E] \) a subset \( S \subseteq V \) of vertices of \( G \) is called a **vertex-cover** if \( (i,j) \in E \Rightarrow \text{either } i \in S \text{ or } j \in S \text{ or both} \).

**Optimization Problem**: Find min. size vertex-cover in \( G \).

**Decision Problem**

*Given* \( G = [V; E] \) and a pos integer \( k \), is there a vertex cover in \( G \) of size \( k \)?

We want to show that the **Decision Problem** \( \text{VERTEX-COVER} \in \text{NP-C} \). The first part of this is to show: \( \text{VERTEX-COVER} \in \text{NP} \)

For this, we need a verification certificate for "Yes" instances and a verification algorithm. That checks that a "Yes" instance is indeed a "Yes" instance. We do this next.
At the given instance of \( \text{VERTEX-COVER} \) is a "yes" instance, then \( \exists S \subseteq V, |S| = k \) that is a vertex cover. \( S \) is the certificate that "we" provide for "yes" instance. [If we attempted to "cheat" and tried to provide such a \( S \) for a "no" instance, the verification alg. will "catch" any attempt !!!] (No FALSE certificate!)

The verification algorithm now checks that \(|S| = k\), and \( \exists (i,j) \in E \) either \( i \in S \) or \( j \in S \) or both.

If this is not the case, alg. returns "Cheat".

Note: Size of certificate is poly in Size of graph and \( V \), alg. is poly. time.

This complete proof of \( \text{VERTEX-COVER} \in \text{NP} \).

Now we move to the second part of this process. To show \( \text{NP} \subseteq \text{P} \), \( \text{NP} \subseteq \text{P} \).

Thumb Rule (may not always work, but do in many cases)

If the problem that you wish to show \( \text{NP} \subseteq \text{P} \)
[ the one on the "right" of \( \leq_p \) ]
is a graph problem, try to use a graph problem on the left; for number problem use number 1.
So far, CIRCUIT-SAT \{ 3-SAT \} \{ 3-SAT \} \text{"logic" problems}

\text{CLIQUE} \text{"graph"}

Later: SUBSET-SUM \text{"Number" problems}

This"trick" (thumb rule) does NOT apply
when you try to show \text{NPC} for the "first"
problem of each kind. For example,
when we did \text{CLIQUE}, there was no
known \text{NPC} problem of the "graph" type.
Here "thumb rule" does NOT apply. There
are cases where even if thumb-rule
can be used, we choose NOT to. This
is because there might be an easier way
to do this with another type. This is
why it is called a "thumb" rule.

Having said all of the above, for \text{VERTEX-COVER}
we will use the thumb rule and
\text{Show CLIQUE} \leq_p \text{VERTEX-COVER}

Note: problem to be shown \text{NPC} is on the right
and known \text{NP-C} problem is on the left.
CLIQUE \leq_p \text{ VERTEX-COVER} \\
\text{(Arbitrary) instance} \quad \{ \\
\text{undir } G = [V; E] \quad \text{and pos.int } k \quad \}
\text{instance} \quad \{ \\
\text{undir } H = [U; F] \quad \text{and a pos.int } l.
\}

\underline{Transformation Process} \\
(Done without knowledge of whether instance \text{ of CLIQUE is yes or no. Transformation is the same for both.})

Def: given an \textit{undir} graph \( G = [V, E] \), its \underline{Complement graph} \( \overline{G} = [V; \overline{E}] \), where \( \overline{E} = \{ (i, j) \in E \mid (i, j) \notin V \} \).

Same set of vertices in both. An edge in one, means that edge is NOT in the other and conversely. Both sets of edges together give all possible edges between pairs of nodes in \( V \).

\( H = \overline{G}, \ l = |V| - k \)

Next we show "Corresponding" part.
If $S$ is a vertex cover in $G$, $(V-S)$ is an independent set (anti-clique) in $G$ and hence a clique in $\overline{G}$.

**Def:** A subset $S \subseteq V$ of nodes is called an independent set (anti-clique) if no two nodes in $S$ are connected by an edge.

**Example**

![Graph with vertex cover and independent set]

- **Green set** is a vertex-cover in $G$.
- **Red set** is an independent set in $G$.
- **Clique**.

**Proof:**

| $S$ is a vertex cover in $G$ of size $k$ | $V-S$ is a clique in $\overline{G}$ of size $|V| - k$. |

Hence, the transformation is correct, so Clique $\leq_p$ Vertex-Cover and hence Vertex-Cover $\in$ NP.