NP-Completeness

• A language $B$ is NP-complete iff
  • $B \in \text{NP}$
  • $\forall A \in \text{NP} \quad A \leq_P B$

This property means $B$ is NP hard
3SAT is NP-complete
Result

Idea:
• B is known to be NP complete
• Use it to prove NP-Completeness of C

IF
• C is NP-hard
  • B is NP-complete and
  • B \leq_p C and
• C \in NP

THEN
• C is NP-complete
Result

• **CLIQUE** is NP-hard
• **3SAT** is NP-complete and
• **3SAT** ≤ₚ **CLIQUE** and
• **CLIQUE** ∈ NP

THEREFORE

• **CLIQUE** is NP-complete
Result

• **VERTEX-COVER** is NP-hard
• **3SAT** is NP-complete and
  • **3SAT** $\leq_P$ **VERTEX-COVER** and
• **VERTEX-COVER** $\in$ NP

**THEREFORE**

• **VERTEX-COVER** is NP-complete
Result

IF
• \( B \) is NP-complete and
• \( B \in P \)

THEN
• \( P = NP \)
SAT $\in$ P $\iff$ P = NP

3SAT $\in$ P $\iff$ P = NP

CLIQUE $\in$ P $\iff$ P = NP

VERTEX-COVER $\in$ P $\iff$ P = NP
\[
\text{SAT} = \{ \langle \Phi \rangle \mid \Phi \text{ is a satisfiable Boolean formula} \}
\]

\[
\text{SAT} \in \text{NP}
\]
Cook-Levin Theorem

\textbf{SAT} is \textbf{NP}-complete

\begin{itemize}
  \item \textbf{SAT} \in \textbf{NP}
  \item \forall \textbf{A} \in \textbf{NP} \quad \textbf{A} \leq_P \textbf{SAT}
\end{itemize}
To prove

\[ \forall A \in \text{NP} \quad A \leq_P \text{SAT} \]

define \text{polynomial time computable} function \( f : \Sigma^* \rightarrow \Sigma^* \)

\[ \forall w \in \Sigma^* \quad w \in A \iff f(w) \in \text{SAT} \]

\[ \forall w \in \Sigma^* \quad w \in A \iff \langle \Phi \rangle \in \text{SAT} \]
Consider any $A \in \text{NP}$

$\exists \text{NTM } N$ that decides $A$ in polytime $n^k$

For any input $w \in \Sigma^*$

valid **tableau** of configurations

*Figure 7.38*

A tableau is an $n^k \times n^k$ table of configurations
Properties of a Valid Tableau

There is exactly one symbol in each cell

The first row is the (‘legal’) start configuration

Every subsequent row is generated ‘legally’
For any input $w \in A$ there is an **accepting** (valid) **tableau** of configurations.
Properties of an Accepting Tableau

There is exactly one symbol in each cell
The first row is the (‘legal’) start configuration
Every subsequent row is generated ‘legally’
One of these rows is an accepting configuration
Given $N$ and $w$ construct a **Boolean formula** that is satisfiable *exactly when* $N$ has an **accepting tableau** on input $w$
Define **Boolean formula** with variables $x_{ijs}$ for
1. $1 \leq i \leq n^k$
2. $1 \leq j \leq n^k$
3. $s \in \text{State Set} \cup \text{Tape Alphabet} \cup \text{Delimiter}$

Want following semantics:

$x_{ijs}$ is $T$ iff cell $(i, j)$ contains symbol $s$

for some valid accepting tableau
$x_{ijs}$ is $T$ iff cell $(i, j)$ contains symbol $s$.

$x_{12q_0}$ is $T$ whereas $x_{12\sqcup}$ is $F$.
Properties of an Accepting Tableau

There is exactly one symbol in each cell
The first row is the (‘legal’) start configuration
Every subsequent row is generated ‘legally’
One of these rows is an accepting configuration

![Diagram of a tableau with symbols and configurations](image)

**Figure 7.38**
A tableau is an $n^k \times n^k$ table of configurations
represent valid tableau with a **Boolean formula** with components

- $\Phi_{\text{cell}}$

- exactly one symbol per cell

\[
\phi_{\text{cell}} = \bigwedge_{1 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \land \left( \bigwedge_{s,t \in C, s \neq t} \overline{x_{i,j,s}} \lor x_{i,j,t} \right) \right]
\]

for any pair $(i,j)$

the cell contains at *least* one symbol

the cell contains at *most* one symbol
Properties of an Accepting Tableau

There is exactly one symbol in each cell.
The first row is the (‘legal’) start configuration.
Every subsequent row is generated ‘legally’.
One of these rows is an accepting configuration.

![Diagram of a tableau](image)

**Figure 7.38**
A tableau is an $n^k \times n^k$ table of configurations.
represent valid tableau with a **Boolean formula** with components

- $\Phi_{\text{cell}}$
  - exactly one symbol per cell
- $\Phi_{\text{start}}$
  - legal starting configuration

\[
\phi_{\text{start}} = x_{1,1},\# \land x_{1,2},q_0 \land x_{1,3},w_1 \land x_{1,4},w_2 \land \ldots \land x_{1,n+2},w_n \land x_{1,n+3,\square} \land \ldots \land x_{1,n^k-1,\square} \land x_{1,n^k,\#}.
\]
Properties of an Accepting Tableau

There is exactly one symbol in each cell.
The first row is the (‘legal’) start configuration.
Every subsequent row is generated ‘legally’.
One of these rows is an accepting configuration.

FIGURE 7.38
A tableau is an $n^k \times n^k$ table of configurations.
represent valid tableau with a **Boolean formula** with components

- $\Phi_{\text{cell}}$
  - exactly one symbol per cell
- $\Phi_{\text{start}}$
  - legal starting configuration
- $\Phi_{\text{move}}$
  - legal moves
represent valid tableau with a **Boolean formula**

with components

- $\Phi_{\text{move}}$
- legal moves

transition function

\[
\begin{array}{c}
\# & a & b & q_1 & b & c & \square & \square & \square & \square & \square & \square \\
\# & a & q_2 & b & c & c & \square & \square & \square & \square & \square & \square \\
\# & a & b & a & q_2 & c & \square & \square & \square & \square & \square & \square \\
\end{array}
\]
represent valid tableau with a **Boolean formula** with components

- $\Phi_{\text{move}}$
- legal moves represented by legal windows

transition function
represent valid tableau with a **Boolean formula** with components

- $\Phi_{\text{move}}$
- legal moves represented by legal windows

\[
\phi_{\text{move}} = \bigwedge_{1 \leq i < n^k, \ 1 < j < n^k} (\text{the } (i, j)\text{-window is legal}).
\]

\[
\bigvee_{a_1, \ldots, a_6} (x_{i, j-1, a_1} \land x_{i, j, a_2} \land x_{i, j+1, a_3} \land x_{i+1, j-1, a_4} \land x_{i+1, j, a_5} \land x_{i+1, j+1, a_6})
\]

is a legal window
Properties of an **Accepting Tableau**

There is *exactly* one symbol in each cell.
The first row is the (‘legal’) start configuration.
Every subsequent row is generated ‘legally’.

One of these rows is an accepting configuration.
represent valid \textit{accepting} tableau with a **Boolean formula** with components

- $\Phi_{\text{cell}}$
  - exactly one symbol per cell
- $\Phi_{\text{start}}$
  - legal starting configuration
- $\Phi_{\text{move}}$
  - legal moves
- $\Phi_{\text{accept}}$
  - legal accepting configuration

\[
\phi_{\text{accept}} = \bigvee_{1 \leq i, j \leq n^k} x_{i,j,q_{\text{accept}}}
\]
represent valid accepting tableau with **Boolean formula**

\[ \Phi_{\text{cell}} \land \Phi_{\text{start}} \land \Phi_{\text{move}} \land \Phi_{\text{accept}} \]
\( \forall w \in \Sigma^* \quad w \in A \)

\( \iff \)

there is a valid accepting tableau

\( \iff \)

constructed formula is SATISFIABLE

Corollary 7.42: \textbf{3SAT} is \textbf{NP-complete}
\[ \text{HALT}_{\text{TM}} = \{ \langle M, w \rangle : M \text{ is a TM that halts on input } w \} \]

\[ \text{HALT}_{\text{TM}} \text{ is NP-hard} \]

In other words
\[ \forall A \in \text{NP} \quad A \leq_{\text{P}} \text{HALT}_{\text{TM}} \]
3SAT $\leq_P$ HALT$_{TM}$

**define** polynomial time **computable** function $f : \Sigma^* \rightarrow \Sigma^*$

$\forall \ w \in \Sigma^* \quad w \in 3SAT \iff f(w) \in \text{HALT}_{TM}$
Define TM $M$ as follows

$M$: On input $w$:

If $w$ is not a valid 3CNF encoding
then loop

If $w$ is a valid 3CNF encoding
then

check if $w$ evaluates to True for any possible assignment to variables in $w$
If yes then accept else loop

$\forall w \in \Sigma^* \quad w \in 3SAT \iff \langle M, w \rangle \in \text{HALT}_{TM}$
\[ \text{HALT}_{TM} = \{ \langle M, w \rangle : M \text{ is a TM that halts on input } w \} \]

Is \( \text{HALT}_{TM} \) NP-complete?
All languages

P

NP

NP-complete

NP-hard