The SAT is NP-hard.

SATisfiability is:

Given a propositional formula
\[ F(U_1, U_2, ..., U_n) = (U_1 \vee \neg U_2 \vee \neg U_3) \land (\neg U_1 \land U_2) \]

Find: Values \( U_i \) that make \( F \) true

Motivation: Check program correctness, to find out a case of going into branching

\[
\text{if } (U_1 \lor U_2 \lor \neg U_3) \land (\neg \text{else } \text{if } \\
\text{SAT is in NP}
\]

NP-hard is any problem from class NP can be reduced to it.

Prove:
That any given NP problem can be reduced to SAT.

NP problem: We have a feasible predicate \( C(x, y) \)

- We have a polynomial \( P(x) \)

Instance: Given \( x \) find \( y \) s.t. \( C(x, y) \) is true and \( \text{len}(y) \leq P(x) \)

Feasible: We have a computational device
that, given $x$ and $y$, checks whether $C(x,y)$ is true in polynomial time

$$t \leq O(\text{len}(x)+\text{len}(y))$$

Reduce to SAT

To every input $x$ of the NP-problem we put into correspondence a propositional formula.

Let's consider the device for checking $c(x,y)$

The device consists of basic parts called cells.

$N$ cells - # of cells.

Moments of time: $0, \Delta t, 2\Delta t$

Each cell has different states. The state of all $i$ at moment $t$ will be denoted by $S_i(t)$

Let $S$ denote max # of states

How does comp. device works?

$$S_i(t+1) = f_{i,t}(S_i(t), \ldots, S_j(t), \ldots)$$

State of all $i$ in next moment of time

$j$ is a neighbor
\( r = c \cdot \Delta t \)

\[ \Delta V \text{ denote the size of the smallest cell} \]

\[ V = \frac{1}{3} \pi r^3 = \frac{1}{6} \pi c^3 \Delta t^3 \]

\[ N_{\text{neigh}} \leq \frac{V}{\Delta V} = \frac{\frac{1}{3} \pi r^3 \Delta t}{\Delta V} \]

We need to describe \( S_{it} \) in terms of \( \theta_i \) and \( \phi_i \)

describe \( f_{it} \) in terms of \( V, k, l \)

\[ s = \{ 0, 0, 0, 1, 1, 0 \} \]

\# of bits \( t_b = \log_2 (s) \)

\( S_{i, b, k} \) - \( b \)-th bit in the description of the state

\[ S_{i, b, t+1} = f_{i, b, t} (S_{i, 1, t}, \ldots, S_{i, b-1, t}, \ldots, S_{j_1, 1, t}, \ldots, S_{j, b, t}) \]

\# of variables is bounded by \( N_{\text{neigh}} \)

\[ \leq N_{\text{neigh}} \]

DNF and CNF forms:

\[ 0.8 \cdot x_1 + 0.8 \cdot x_2 \geq x_3 \]

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( x_3 )</th>
<th>( F )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

\( \text{Disjunction} \)
\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & F \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

\[
\text{Disjunctive Normal Form}
\]

\[
F = (\overline{X_1} \& \overline{X_2} \& \overline{X_3}) \lor (X_1 \& \overline{X_2} \& \overline{X_3}) \lor \( \overline{X_1} \& X_2 \& \overline{X_3}) \lor (X_1 \& X_2 \& \overline{X_3}) \lor (X_1 \& \overline{X_2} \& X_3)
\]

\[
\overline{X_1} \cdot \overline{X_2} \cdot \overline{X_3} + \overline{X_1}
\]

\[\leq 2 \text{ Negh.} \rightarrow B\]

\[
\text{CNF}
\]

\[\text{Conjunction} \quad \land\]

\[\text{Disjunction} \quad \lor\]

First we find CNF form for \( F \)
- We use de Morgan Laws.

Negation:
\[
\begin{array}{cccc|c}
X_1 & X_2 & X_3 & F & \overline{F} \\
\hline
0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 \\
\end{array}
\]

\[
\overline{F} = (\neg X_1 \lor \neg X_2 \lor X_3) \lor (\neg X_1 \lor X_2 \lor \neg X_3) \lor (X_1 \lor \neg X_2 \lor \neg X_3)
\]

\[
F = \neg (\overline{F}) = (X_1 \lor \neg X_2 \lor \neg X_3) \land (X_1 \lor \neg X_2 \lor X_3) \land (\neg X_1 \lor X_2 \lor \neg X_3)
\]

\[
F_i, b, t \equiv \text{CNF form of } S_i, b, t = S_i, b, t(t\ldots)
\]

What does it mean the computer works correctly?

\[
F_i, i_1, \ldots, i_{n-1}, 0 \ldots 0, F_{i, b_1} \land \ldots \land F_{i, b_{n-1}} \land \bigwedge_{i \in \text{Input}} X_i
\]

\[
(F_i = X_i) \land (F_{i, b_1} = X_2) \land (F_{i, b_{n-1}} = \text{true})
\]
\[ x \sim y \]

\[ X \Rightarrow Y \]

\[ x \sim x' \]

\[ x' \sim y' \]

\[ F_{i, b, t} \quad \text{...} \quad F_{N_{\text{cell}}, l, b, t} \quad \text{...} \]

\[ \text{length}(x') \leq C \cdot N_{\text{cell}} \cdot B \cdot T \]

\[ T \leq P \left( \text{len}(x) + \text{len}(y) \right) \leq P \left( \text{len}(x) + P \text{C}(\text{len}(x)) \right) \]

\[ N_{\text{cell}} \leq \frac{1}{3} \Delta V \cdot c^3 + \frac{1}{3} \Delta V \]

Input: bit \( x \)
Output: bit \( y \)
\( c(x, y) = x \equiv y \)

Property: \( x = y \)

How do we check whether \( x = y \)?
\[ t = 2 \]

\[ t = 3 \quad C(x|y) \quad x = y \]

\[ N_{\text{cells}} = 3 \]

<table>
<thead>
<tr>
<th>x-cell</th>
<th>1 states 01</th>
</tr>
</thead>
<tbody>
<tr>
<td>y-cell</td>
<td>2</td>
</tr>
<tr>
<td>wire</td>
<td>3</td>
</tr>
</tbody>
</table>

3 states:
- Sending 0 00
- Sending 1 01
- No transmitting 10

\[ S_{112} = S_{111} \]
\[ S_{211} = S_{211} \]
\[ S_{311} = 0 \]

Subject:

6-th bit in the description of
\[ S_{3,1,2} = S_{2,1,1} \]  
the description of  
1-th cell at moment \( t \)