SAT is NP-complete

Cook showed that the language SAT of satisfiable Boolean expressions is NP-complete.

To establish this, we need to show that for every language $L$ in NP, there is a polynomial time reduction from $L$ to SAT.

Since $L$ is in NP, there is a nondeterministic Turing machine

$$M = (K, \Sigma, s, \delta)$$

and a bound $n^k$ such that a string $x$ is in $L$ if, and only if, it is accepted by $M$ within $n^k$ steps.

Intuitively, these variables are intended to mean:

- $S_{i,q}$ – the state of the machine at time $i$ is $q$.
- $T_{i,j,\sigma}$ – at time $i$, the symbol at position $j$ of the tape is $\sigma$.
- $H_{i,j}$ – at time $i$, the tape head is pointing at tape cell $j$.

We now have to see how to write the formula $f(x)$, so that it enforces these meanings.

Boolean Formula

We need to give, for each $x \in \Sigma^*$, a Boolean expression $f(x)$ which is satisfiable if, and only if, there is an accepting computation of $M$ on input $x$.

$f(x)$ has the following variables:

- $S_{i,q}$ for each $i \leq n^k$ and $q \in K$
- $T_{i,j,\sigma}$ for each $i, j \leq n^k$ and $\sigma \in \Sigma$
- $H_{i,j}$ for each $i, j \leq n^k$

Initial state is $s$ and the head is initially at the beginning of the tape.

$$S_{1,s} \land H_{1,1}$$

The head is never in two places at once

$$\bigwedge_{i,j} (H_{i,j} \land \neg H_{i,j'})$$

The machine is never in two states at once

$$\bigwedge_{q,i} (S_{i,q} \land \neg S_{i,q'})$$

Each tape cell contains only one symbol

$$\bigwedge_{i,j,\sigma} (T_{i,j,\sigma} \land \neg (T_{i,j,\sigma'}))$$
The initial tape contents are $x$

$$\bigwedge_{j \leq n} T_{1,j} \cdot x_j \land \bigwedge_{n < j} T_{1,j,\bot}$$

The tape does not change except under the head

$$\bigwedge_{i} \bigwedge_{j} \bigwedge_{j' \neq j} \bigwedge_{\sigma}(H_{i,j} \land T_{i,j',\sigma}) \rightarrow T_{i+1,j',\sigma}$$

Each step is according to $\delta$.

$$\bigwedge_{i} \bigwedge_{j} \bigwedge_{\sigma} \bigwedge_{q}(H_{i,j} \land S_{i,q} \land T_{i,j,\sigma}) \rightarrow \bigvee_{\Delta}(H_{i+1,j'} \land S_{i+1,q'} \land T_{i+1,j',\sigma'})$$

where $\Delta$ is the set of all triples $(q', \sigma', D)$ such that $(q, \sigma, (q', \sigma', D)) \in \delta$ and

$$j' = \begin{cases} j & \text{if } D = S \\ j - 1 & \text{if } D = L \\ j + 1 & \text{if } D = R \end{cases}$$

Finally, some accepting state is reached

$$\bigvee_{i} S_{i,\text{acc}}$$

---

**CNF**

A Boolean expression is in **conjunctive normal form** if it is the conjunction of a set of **clauses**, each of which is the disjunction of a set of **literals**, each of these being either a **variable** or the **negation** of a variable.

For any Boolean expression $\phi$, there is an equivalent expression $\psi$ in conjunctive normal form.

$\psi$ can be exponentially longer than $\phi$.

However, **CNF-SAT**, the collection of satisfiable **CNF** expressions, is **NP**-complete.

---

**3SAT**

A Boolean expression is in **3CNF** if it is in conjunctive normal form and each clause contains at most 3 literals.

**3SAT** is defined as the language consisting of those expressions in **3CNF** that are satisfiable.

**3SAT** is **NP**-complete, as there is a polynomial time reduction from **CNF-SAT** to **3SAT**.
Composing Reductions

Polynomial time reductions are clearly closed under composition. So, if $L_1 \leq_p L_2$ and $L_2 \leq_p L_3$, then we also have $L_1 \leq_p L_3$.

Note, this is also true of $\leq_L$, though less obvious.

If we show, for some problem $A$ in NP that

$$SAT \leq_p A$$

or

$$3SAT \leq_p A$$

it follows that $A$ is also NP-complete.