Objective

- Exhibit a problem that belongs to \textit{NP-complete}
The satisfiability problem

- $X = \{x_1, x_2, \ldots, x_n\}$: set of variables
- $C = \{C_1, C_2, \ldots, C_m\}$: set of clauses
- $\mathcal{F} = C_1 \land C_2 \land \ldots \land C_m$

$SAT = \{\langle \mathcal{F} \rangle \mid \mathcal{F} \text{ is a satisfiable Boolean formula} \}$
Cook-Levin theorem

**Theorem**

*(original formulation)* $\text{SAT} \in \mathcal{P} \text{ if and only if } \mathcal{P} = \mathcal{NP}$.

equivalently: SAT is **NP-complete**.
$\text{SAT} \in \text{NP-complete}$

$\text{SAT} \in \mathcal{NP}$

Informally, given an assignment of variables, we scan all clauses to check if they evaluate to $\text{TRUE}$

The verifier:

1. Generate non-deterministically an interpretation function

2. Evaluate this function and if it is $\text{TRUE}$ then accept

The cost is in $\Theta(n)$
The idea behind the reduction

Coding the execution of a NDTM by means of a boolean expression.

More precisely, we transform any language of \( \mathcal{NP} \) to the encoding of the positive instances of SAT. We extend here the transformation to a pair \((\text{word}, \text{language})\)^1

- We exhibit such a transformation (reduction) that associates to each word \( \omega \) and to each language \( L \) of \( \mathcal{NP} \) an instance of SAT that is positive iff \( \omega \in L \).

The technical point is to show how to code the data related to \( L \): all the informations on the tape and also all the states and transitions.

Then, we will verify that it is polynomial in \(|\omega|\) for any fixed language.

---

^1usually, it is restricted to a word
How to proceed?

The only characterization of the languages in $\mathcal{NP}$ is to be accepted by a non-deterministic Turing Machine.

Thus, we will build such a transformation:

- given a NDTM $M (K, \Sigma, \Gamma, \Delta, \text{start}, \text{halt})$ and an input word $\omega$, it produces a positive instance of SAT iff $M$ accepts $\omega$
Express the execution as a SAT formula

\[ A \leq_p \text{SAT} \text{ for every language } A \in \mathcal{NP} \]

- \( M \): a Non-Deterministic Turing Machine that decides \( A \) in polynomial time, say \( n^k \)

- Each configuration is described by a state, the position of the header and the content of the tape.

- An execution is thus fully described by three vectors/table.
Express the execution as a SAT formula

- **Tape:** create a table $T[i][j]$ of size $n^k \times n^k$, we donot count the initial state of the tape at row ”0”
  - each row $T[i]$ corresponds to a configuration of the tape
- the head is recorded into a vector called $P$
- the current state is on vector $Q$
- An extra vector is introduced for the (non-deterministic) choice of the transition
- a table is **accepting** if any row is an accepting configuration

We introduce the variable $x_{ij,s}$ that means that the symbol $s$ is in $T[i][j]$
SAT ∈ NP-complete

For each $i, j, s$, where $1 \leq i, j \leq n^k$ and $s \in \Gamma \cup \Sigma$, define a variable

$$x_{i,j,s} = \begin{cases} \text{TRUE} & \text{if the cell in row } i \text{ and column } j \text{ contains the symbol } s \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Define clauses to guarantee the calculation of $M$
SAT \in \text{NP-complete}

For each \(i, j, s\), where \(1 \leq i, j \leq n^k\) and \(s \in \Gamma \cup \Sigma\), define a variable

\[
x_{i,j,s} = \begin{cases} 
\text{TRUE} & \text{if the cell in row } i \text{ and column } j \text{ contains the symbol } s \\
\text{FALSE} & \text{otherwise}
\end{cases}
\]

Define clauses to guarantee the calculation of \(M\)

- there is exactly one symbol in each cell

\[
\phi_{\text{cell}} = \bigwedge_{0 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in \Gamma \cup \Sigma} x_{i,j,s} \right) \land \left( \bigwedge_{s,t \in \Gamma \cup \Sigma} \left( \bar{x}_{i,j,s} \lor \bar{x}_{i,j,t} \right) \right) \right]
\]
We verify similarly that:

- there is a unique configuration
- the header is pointing only a unique cell
- there is only one choice for a transition

These conditions are expressed as boolean CNF expressions.
Other conditions

- the first row corresponds to the starting configuration
- each configuration is obtained from the previous one by a transition $c_i \vdash_M c_{i+1}$
- there is an accepting state before $n^k$ steps

Can you write these conditions as CNF expressions?
The first row corresponds to the starting configuration

The word $\omega$ is on the tape and all other cells are filled by $\square$. The header should be at the left of this input word. The process starts at the first state.

$$\phi_{\text{init}} = \left[ \left( \bigwedge_{0 \leq j \leq n-1} x_{0,j,\omega_{n+1}} \right) \land \left( \bigwedge_{n \leq j \leq n^k} \left( x_{0,j,\square} \right) \right) \right] \land p_{0,0} \land q_{0,\text{start}}$$

- This boolean expression is in CNF and its length is in $O(n^k)$
There is an accepting state

- there is an accepting state

\[ \phi_{\text{accept}} = \bigvee_{1 \leq i \leq n^k} q_{i,\text{halt}} \]
The hardest part

Proving that the successive configurations are valid (it is in accordance with the transition table).

This is done through two conditions

- All the cells of the tape that are not concerned by the header are not modified.
- The transformation from a step to the next is valid.
The not concerned cells are unchanged

\[
\phi_{\text{move}} = \bigwedge_{0 \leq i, j < n^k, s \in \Gamma \cup \Sigma} \left[ (x_{i,j,s} \land \bar{p}_{i,j}) \Rightarrow x(i+1), j, s \right]
\]

that can be written as a CNF as follows

\[
\phi_{\text{move}} = \bigwedge_{0 \leq i, j \leq n^k, s \in \Gamma \cup \Sigma} \left[ \bar{x}_{i,j,s} \lor p_{i,j} \lor x(i+1), j, s \right]
\]
The not concerned cells are unchanged

\[ \phi_{\text{move}} = \bigwedge_{0 \leq i,j < n^k, s \in \Gamma \cup \Sigma} \left[ (x_{i,j,s} \land \overline{p}_{i,j}) \Rightarrow x_{(i+1),j,s} \right] \]

that can be written as a CNF as follows

\[ \phi_{\text{move}} = \bigwedge_{0 \leq i,j \leq n^k, s \in \Gamma \cup \Sigma} \left[ \overline{x}_{i,j,s} \lor p_{i,j} \lor x_{(i+1),j,s} \right] \]

The last condition is left to the reader.
SAT $\in$ NP-complete

Construct $\mathcal{F} = \phi_{\text{cell}} \land \phi_{\text{start}} \land \phi_{\text{accept}} \land \phi_{\text{move}}$

- $\mathcal{F}$ has $O(n^k)$ variables and clauses
\[ \text{SAT } \in \text{NP-complete} \]

Construct \( \mathcal{F} = \phi_{\text{cell}} \land \phi_{\text{start}} \land \phi_{\text{accept}} \land \phi_{\text{move}} \)

- \( \mathcal{F} \) has \( O(n^k) \) variables and clauses

**Theorem:** \( \mathcal{F} \) is satisfiable if and only if \( A \) is decided by \( M \)