"I can't find an efficient algorithm, I guess I'm just too dumb."
"I can't find an efficient algorithm, because no such algorithm is possible!"
"I can't find an efficient algorithm, but neither can all these famous people."
\text{Sat}\in\text{NP}: \text{proof} = \text{satisfying assignment}

\text{Graph coloring}\in\text{NP}: \text{proof} = \text{coloring}

\text{k-clique}, \text{k-vertex cover}, \text{k-independent set}\in\text{NP}

\text{Tautology}\in\text{co-NP} \text{ ("no" instances have a proof)}
NP complete:

1) in NP
2) every problem in NP reducible to it.

Transitivity of p-time reduction implies

NP complete iff

1) in NP
2') some NP-complete problem reducible to it.

We need one NP-complete problem to get started.
Cook–Levin Theorem: SAT is NP-complete.

Given any p-time verifier, construct (in p-time) an instance of SAT s.t. verifier answers "yes" iff formula is satisfiable.

Verifier: Turing Machine

In one step, machine can write a symbol, move head one position, change state.

What to do is based on state, symbol read.

Fixed # of states, fixed # of tape symbols, including blank; start state, "yes" state, ("no" state)

Explicitly given polynomial time bound p(n).
Input (of size $n$) is a "yes" instance iff for some "proof" and given input, the machine reaches "yes" state within $p(n)$ steps from start state.

Must construct a formula that is satisfiable iff this happens.

Note: input is specified, proof is not (non-deterministic part)

Proof can't exceed length $p(n)$: machine can't get farther in $p(n)$ steps.

Can assume machine loops in "yes" state; if ever in "yes", will be in "yes" at step $p(n)$.
$States: \ 1, \ldots, y \quad 1 = start, \ y = yes$

$Symbols: \ 1, \ldots, z \quad 1 = blank$

$Tape \ cells, \ -p(n), \ldots, 0, \ldots, p(n)$

$Time: \ 0, 1, \ldots, p(n)$

$Variables \ for \ formula:$

$hit: \ true \ if \ head \ on \ tape \ cell \ i \ at \ time \ t$

$-p(n) \leq i \leq p(n), \ 0 \leq t \leq p(n)$

$s_{j,t}: \ true \ if \ state \ j \ at \ time \ t$

$1 \leq j \leq y, \ 0 \leq t \leq p(n)$

$c_{i,k,t}: \ true \ if \ tape \ cell \ i \ holds \ symbol \ k \ at \ time \ t$

$-p(n) \leq i \leq p(n), \ 1 \leq k \leq z, \ 0 \leq t \leq p(n)$
What does the formula need to say?

At most one state, head position, and symbol per cell at each time:

\[( \overline{h_{i,t}} \lor \overline{h_{i',t}} ) \quad i \neq i', \text{ all } t \]

\[( \overline{s_{j,t}} \lor \overline{s_{j',t}} ) \quad j \neq j', \text{ all } t \]

\[( \overline{c_{ik,t}} \lor \overline{c_{ik',t}} ) \quad k \neq k', \text{ all } i, \text{ all } t \]

Correct initial state, head position, and tape contents:

\[h_0 \land s_0 \land s_{10} \land c_{010} \land c_{1k_0} \land c_{2k_0} \land \cdots \land c_{nk_0} \]

\[\land c_{(n+1)10} \land \cdots \land c_{r(n+1)0} \]

Input is \(k_1 k_2 \cdots k_n\), rest of right side of tape is blank.
Correct final state: \( s_y p(n) \)

Correct transitions:

E.g. if machine in state \( j \) reads \( k \), it then writes \( k' \), moves head right, and changes to state \( j' \):

\[
\begin{align*}
&\text{s}_{j t} \land \text{h}_{it} \land \text{c}_i \text{k}_t = \text{s}_{j' t+1} \land \text{h}_{i+1 t+1} \land \text{c}_i \text{k'}_{t+1} \\
&\hspace{1cm} (\Rightarrow = "\text{implies}" ) \hspace{1cm} (\text{for each } j, k, t) \\
&\text{h}_{it} \land \text{c}_i \text{k}_t = \text{c}_i \text{k'}_{t+1} \hspace{1cm} (\text{for } i \neq i', \text{each } k, t) \\
&\hspace{1cm} (\text{unread tape cells are unaffected})
\end{align*}
\]

\( \text{CNF?} \)
Any proof that gives a "yes" execution gives a satisfying assignment, and vice-versa.

Conclusion: SAT is NP-complete

(and k-coloring, k-clique, k-independent set, k-vertex cover)
\((x \land y \land z) \subseteq (a \land b \land c)\)

\(\Rightarrow\)

\((\neg x \lor y \lor z \lor \neg a)\)

\(\land\)

\((\neg x \lor y \lor z \lor \neg b)\)

\(\land\)

\((\neg x \lor y \lor z \lor \neg c)\)

\(\Rightarrow\)

\((\neg x \lor \neg y \lor \neg z \lor \neg a)\)

\(\land\)

\((\neg x \lor \neg y \lor \neg z \lor \neg b)\)

\(\land\)

\((\neg x \lor \neg y \lor \neg z \lor \neg c)\)
Subset Sum is NP-complete

Given n integers, and a target k, is there a subset that sums to exactly k?

\[
\{2, 5, 6, 8, 9, 12\} \quad k = 31
\]

yes: 5, 6, 8, 12 \quad n = \text{no fits}

(\text{no for } k = 30)

In NP: subset is proof (verifiable in p-time)

Some NPC problem reducible to subset sum

reduce 3-CNF sat to subset sum
Write numbers base 10

\[(x \lor \neg y \lor \neg z) \land (\neg x \lor y \lor z) \land (y \lor z)\]

\[
\begin{array}{ccccccc}
\neg x & x & y & z & c_1 & c_2 & c_3 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 0 \\
\hline
1 & 1 & 1 & 1 & 4 & 4 & 4 \\
\end{array}
\]

Required sum \(= k\)

Interpret each row as a base 10 number.

Subset sum has a solution iff formula is satisfiable.