NP-completeness

2 perspectives:

1. To show that some problems are (probably) hard to solve
2. A rich topic in theoretical computer science

To show a language is in NP:

2-input verification algorithms with polynomial running time

Examples:

\[ L_{HAM} = \{G \mid G \text{ has a Hamiltonian cycle}\} \]
\[ L_{COMPOSITE} = \{n \mid n \text{ is a composite number}\} \]

What about

\[ L_{PRIME} = \{n \mid n \text{ is a prime number}\} \]?
\[ L_{NOT-HAM} = \{G \mid G \text{ does not have a Hamiltonian cycle}\} \]

A language \( L \) is in the complexity class co-NP if its complement is in NP.
NP-completeness

The circuit satisfiability problem CIRCUIT-SAT

A combinatorial circuit of and/ or/ not gates

Represented as a directed acyclic graph

$L_{\text{CIRCUIT-SAT}} = \{ C | C \text{ is a satisfiable combinatorial circuit} \}$

Lemma 34.5. $L_{\text{CIRCUIT-SAT}}$ is in NP

Lemma 34.6. $L_{\text{CIRCUIT-SAT}}$ is NP-hard

Proof - the standard reduction

Let $L$ be any language in NP, accepted by the 2-input verification algorithm $A(x,y)$

Given any input $x$ (Is $x$ in $L$?)

-Compute $f(|x|)$, where $f(|x|)$ is the running time of $A(x,y)$

-Make $f(|x|)$ copies of the comb circuit of a computer, and $f(|x|)+1$ copies of the memory of the computer, and wire the copies together

-Initialize the first memory copy to $A$, $x$, and $y$

-Ignore all bits except the output bit on the $f(|x|)+1$ ‘th memory copy
Cook (1971). The complexity of theorem-proving procedures