Millennium Prize Problems

Seven famous problems in math stated in 2000 by the Clay Foundation $1,000,000 prize for solving any of them

One of the problems: P vs. NP

Polynomial Time Complexity

Is there a fixed constant c and an algorithm A such that A solves the decision problem in time $O(n^c)$?

Verifying solutions

In some problems (like Sudoku), verifying the solution can be done efficiently

$NP = \text{Decision problems whose solutions can be verified in polynomial time in their input size}$

The N in NP stands for "nondeterministically"

Here’s how P vs. NP is usually (informally) stated:

Let $L$ be an algorithmic task.

Suppose there is an efficient algorithm for verifying solutions to $L$. "$L \in NP$"

Is there always also an efficient algorithm for finding solutions to $L$? "$L \in P$"
Definition of P

An input is encoded as a binary string.

\[ P = \{ L \subseteq \{0, 1\}^* \mid \exists \text{ polynomial time algorithm} \text{ for deciding } L \} \]

Definition of NP

\[ NP = \{ \{ L \subseteq \{0, 1\}^* \mid \exists \text{ polynomial time verifier } \} \text{ for L} \text{ true, where } x \in L \text{ and } |y| \leq O(|x|) \} \]

Definition of NP-hard

\[ \text{NP-hard} = \{ L \subseteq \{0, 1\}^* \mid \forall X \in \text{NP and } X \leq_p L \} \]

To reduce problem X to problem L (we write \( X \leq_p L \)) we want a function f that maps X to L such that:
1) f is polynomial time computable
2) \( x \in X \) if and only if \( f(x) \in L \).

In short. We need to convert X into L.

Lemma. If \( A \leq_p B \) and \( B \in \text{P} \) then \( A \in \text{P} \).

Definition of NP-complete

L is NP-complete iff
1) \( L \subseteq \text{NP} \)
2) \( L \subseteq \text{NP-hard} \)
   
2) For all \( Y \subseteq \text{NP} \), \( Y \leq_p L \)

Venn Diagram (\( P \neq \text{NP} \))
**NP-complete Reduction**

A recipe for proving any \( L \in \text{NP-complete} \):

1) Prove \( L \in \text{NP} \)
2) Choose \( A \in \text{NPC} \) and reduce it to \( L \)
2.1) Describe mapping \( f : A \rightarrow L \)
2.2) Prove \( x \in A \iff f(x) \in L \)
2.3) Prove \( f \) is polynomial

**Conjunctive Normal Form**

Let \( X_k \) denote variables.
We define literals as either \( X_k \) or \( \neg X_k \).

The conjunctive normal form (CNF) is an AND of OR clauses. For example,

\[(X_1 \lor X_2 \lor \neg X_3) \land (X_1 \lor \neg X_2 \lor X_4) \land \ldots\]

**SAT Problem**: is there exist a set of variables that satisfy a given CNF?

**Cook-Levin Theorem (1971)**

SAT is \( \text{NP-complete} \)

No proof, see Kozen’s textbook.

**3-CNF problem (or 3-SAT)**

Each clause has a most 3 literals.

**Question**: Is there such a set of input variables that 3-cnf is true?

**Theorem**: 3-CNF is \( \text{NP-complete} \)

**Proof**.

3-CNF \( \subseteq \text{NP} \)
We need to show CNF \( \leq_p \) 3-CNF.

**CNF \( \leq_p \) 3-CNF**

We need to convert any CNF into 3-CNF...

**Claim**:
\[(a \lor b \lor c \lor d) \text{ is true iff } (a \lor b \lor x) \land (\neg x \lor c \lor d) \text{ is true}\]

\[(a \lor b \lor c \lor d \lor e) \text{ converts to } (a \lor b \lor x) \land (\neg x \lor c \lor y) \land (\neg y \lor d \lor e)\]

The rest of the proof is left as an exercise to a reader.

**Clique is NP-complete**

1) Clique is in \( \text{NP} \)
2) We will show that SAT \( \leq_p \) Clique
Create a vertex for each variable in a clause, assume k-clauses.

Two vertices (from different clauses) are connected if one is NOT negation of other.

A CNF is satisfiable if at least one literal in each clause is true. Thus those literals create a k-clique.
Sudoku

Theorem (2002)

There is a polynomial reduction from 3-coloring to sudoku.

Complexity universe

All of these problems poly-reduce to one another!

Travelling Salesman is an intellectual thriller about four mathematicians hired by the U.S. government to solve the most elusive problem in computer science history — P vs. NP.

The four have jointly created a "system" which could be the next major advancement for our civilization or destroy the fabric of humanity.

www.travellingsalesmanmovie.com/