- Algorithm complexity theory summary
  - The algorithm as a solution to a problem
  - Some algorithms are more efficient than others—complexity theory
  - Some problems have no efficient algorithms to solve them

\[ O(N^k) \]

\( \text{input size, } K \text{ constant} \)

- Some problems have no solutions at all

problem complexity, NPC proof

Note Title 11/3/2015
problems & algorithms

The complexity of a problem is the complexity of the best algorithm for solving this problem.

- Finding the max of an unsorted array
- Negative-weight cycle detection

Class $P$: polynomial time

A problem in $P$ can be solved in time polynomial to the input size $O(n^k)$

$NP$: non-deterministic polynomial

A set of problems that are verifiable in polynomial time.

Given a “certificate” of a solution, we could verify that the
Certificate is correct in time polynomial to the size of the problem input.

E.g. In a graph $G$, is there a path from $u$ to $v$ with path weight $k$?

$u \xrightarrow{} \xrightarrow{} v$

$p \subseteq np$

NPC — non-deterministic polynomial complete

- NPC is a class of decision problem. A decision problem $L$ is in NPC if
  1. it is in the set of NP so that any given solution can be verified in polynomial time.
  2. $L$ is in the set of NP-hard.

NPC are the hardest problems in NP.
NP: ① If you establish a problem as NPC, you provide good evidence for its intractability.
② If any problem in NPC can be solved in polynomial time, then
every problem in \( \text{NP} \) can be solved in polynomial time, i.e. \( P = \text{NP} \).

Some problems:

1. The circuit satisfiability problem: "Given a boolean combinational circuit composed of AND, OR, and NOT gates, is it satisfiable?"

\[
x \rightarrow \overline{x} \quad y \rightarrow \overline{z} \quad x \rightarrow \overline{z} \quad z = x \lor y
\]

\[
z = x \land y
\]