Faculty of Science and Technology
Tokushima University

Computational complexity theory

Associate Professor Shin-ichi NAKAYAMA

Time-Complexity

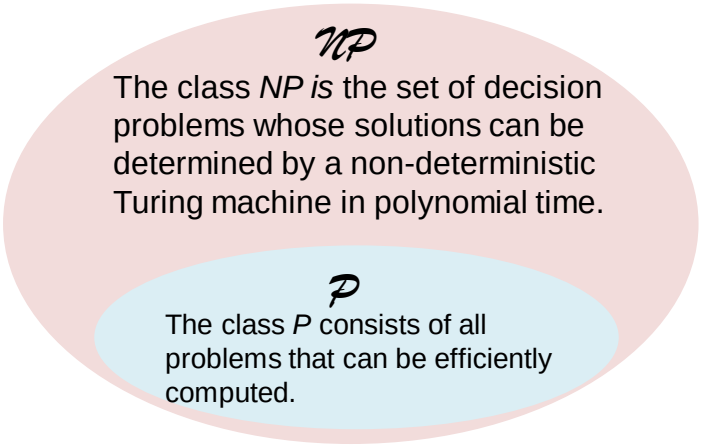
Polynomial Time | **Exponential Time**

$O(\log n)$ $O(n)$ $O(n^2)$ $O(n^k)$ | $O(2^n)$ $O(2^{2n})$ $O(2^{n^2})$

(n is the size of input data and k is the constant number.)

The most fundamental classification is the distinction between problems whose growth rate in terms of time is polynomial and problems whose growth rate is exponential.

P ≠ NP Problem



The P ≠ NP problem is whether *P* and *NP* are in fact the same.

Content:

Computational complexity theory is a branch of the theory of computation in theoretical computer science that focuses on classifying computational problems according to their inherent difficulty, and relating those classes to each other. One aspect of computational complexity is related to an *algorithm* for solving instances of a *problem*. The computational complexity of an algorithm is a measure of how many steps the algorithm will require in the worst case for an instance or input of a given size. The number of steps is measured as a function of that size. Moreover, the theory of computational complexity involves classifying problems according to their inherent tractability or intractability, that is, whether they are "easy" or "hard" to solve. This classification scheme includes the well-known classes *P* and *NP*; the terms "*NP*-complete" and "*NP*-hard" are related to the class *NP*.

In our research, given a problem, we clarify which class it is belong to, and develop an efficient algorithm for solving it if it is belong to the class *P*.

Keywords: computational complexity, algorithm

E-mail: shin@tokushima-u.ac.jp
Tel. +81-88-656-7223
Fax: +81-88-656-7223