Computational Complexity, P and NP – Lecture 16

James Marshall

Complexity Notation Recap – Asymptotic Bounds

$f(n) \in \Omega(g(n)) \Rightarrow$

$f(n) \in O(g(n)) \Rightarrow$

$f(n) \in \Theta(g(n)) \Rightarrow$

Complexity Examples

'Good' complexities

'Bad' complexities

Definition

Let *t(n)* be the running time of an algorithm on input size *n*, defined by
Define the *time complexity class* $\text{TIME}(t(n))$ to be the collection of all languages decidable by an $O(t(n))$ Turing Machine.

Definition

The complexity class **P** is the class of languages that are decidable in polynomial time by a deterministic, single-tape Turing Machine:

$$\mathrm{P} = \bigcup_{k} \mathrm{TIME}\left(n^{k}\right)$$

P *Examples*

Context-Free Languages

EULERIANPATH

PRIMES

Definition

The running time of a non-deterministic TM for deciding some language is the function

where for input size n, $f(n)$ is the length of the longest branch of its computation.

Definition

The time complexity class $\mathrm{NTIME}(t(n))$ is the collection of all languages decidable by an $O(t(n))$ nondeterministic Turing Machine.

Theorem

A non-deterministic TM whose running time is polynomial in some input size, will require an amount of time exponential in that input size in order to simulate on a deterministic TM.

Definition

The complexity class **NP** is the class of languages that are decidable in polynomial time by a nondeterministic Turing Machine:

$$\mathrm{NP} = \bigcup_k \mathrm{NTIME}\left(n^k\right)$$

Definition

A *verifier* for a language A is an algorithm V where

$$A = \left\{ w \middle| V \text{ accepts } \langle w, c \rangle \text{ for some string } c \right\}$$

Definition

A polynomial-time verifier runs (on a deterministic TM) in time polynomial in the size of w.

Definition

The complexity class **NP** is the class of languages that have polynomial time verifiers.

P	NP

NP *Examples*

HAMILTONIANPATH

PRIMEFACTORS

TSP