21.04.2020

21.

> P Class / Polynomial Class Problems

The class of decision problems that can be solved in polynomial time by deterministic algorithms, ie, they can be solved in $O(n^k)$ time in worst case, where $k$ is constant.

Decision problems are problems with yes/no answers.

eg: $O(1)$ – Constant
$O(\log n)$ – Sub-linear
$O(n)$ – Linear
$O(n \lg n)$ – Nearly linear
$O(n^2)$ – Quadratic.

> NP Class / Non-deterministic Polynomial Class Problems.

The class of decision problems that can be solved in polynomial time by non-deterministic algorithms ("Lucky" / magic algo that makes a right guess among the given set of choices).

eg:- Graph coloring
- Hamiltonian cycle
- " path
- Job scheduling with penalties
- Bin packing
- Subset sum problem
- Satisfiability "
- Traveling sales person problem.

Non-deterministic Algorithm has 2 phases and 1 o/p step:

(i) Non-deterministic guessing phase –
- Some completely arbitrary string of chars s, is written beginning at some designated place in memory.
- Each time the algorithm is run, the string written may differ.

(ii) Deterministic verifying phase –
- A deterministic subroutine begins execution.
- In addition to the decision problem's input, the subroutine may use or ignore s.

- Eventually, it returns a value true or false - or it may get in an infinite loop and never halt.

(iii) Output step - If the verifying phase returned true, the algorithm outputs yes.

- the NP class consists of those problems that are verifiable in polynomial time. NP is the class of decision problems for which it is easy to atleast check the correctness of a claimed answer.
- Every problem in this class can be solved in exponential time using exhaustive search.

P vs NP

- Every decision problem that is solvable by a deterministic polynomial time algorithm is also solvable by a polynomial time non-deterministic algorithm.
- All probs in P can be solved using polynomial time algorithms, whereas all problems in NP-P are intractable.

Millenium Prize Million dollar question: Is P = NP?

$P \subseteq NP$ : any problem that can be solved quickly by a computer can also have a particular possible answer that can be quickly checked by the computer.

The reverse - whether or not $NP \subseteq P$ - is unknown: we don't know whether or not probs that have a good algorithm for checking answers also have good algorithms for finding answers.