7 Difference Between P And NP Problems In Computer Science

What Are P And NP Problem?

Computer algorithms take a certain amount of time to solve the problem they are tasked with. Generally, you can roughly estimate how much time an algorithm will take using the number of elements they need to handle. Computer scientists call the number of elements *N*.

Because some algorithms are more or less efficient than others, the time they take to complete could be related to N, N^2, N^3, and so on. The important thing, though, is that the exponent is a constant—it's 1, or 2, etc. When this is the case, an algorithm is said to complete in polynomial time, or *P*.

Unfortunately, not all problems work this way. Solving some problems can take an algorithm an amount of time proportional to 2^N, 3^N, and so on. In this case, N is the exponent, meaning that every element the algorithm has to deal with increases its complexity exponentially. In this case, the algorithm can be completed in exponential time, or *NP* (which really stands for nondeterministic polynomial time).

The difference between these two can be huge. If a *P* algorithm has 100 elements, and its time to complete working is proportional to N^3, then it will solve its problem in about 3 hours. If it's an *NP* algorithm, however, and its completion time is proportional to 2^N, then it will take roughly 300 quintillion years.

Therefore, A problem is called NP if its solution can be guessed and verified in polynomial time, and nondeterministic means that no particular rule is followed to make the guess. On the other hand, a P problem is one that can be solved in polynomial time by deterministic algorithms.

Facts About P Problems

- P problems are set of problems which can be solved in polynomial time by deterministic algorithms.
- The problem belongs to class P if it's easy to find a solution for the problem.
- P Problems can be solved and verified in polynomial time.
- P problems are subset of NP problems.
- It is not known whether P=NP. However, many problems are known in NP with the property that if they belong to P, then it can be proved that P=NP.
- All P problems are deterministic in nature.
- Example: Selection sort, linear search

Facts About NP-Class Problems

- NP problems are the problems which can be solved in non-deterministic polynomial time.
- The problem belongs to NP, if it's easy to check a solution that may have been very tedious to find.
- Solution to NP problems cannot be obtained in polynomial time, but if the solution is given, it can be verified in polynomial time.
- If P≠NP, there are problems in NP that are neither in P nor in NP-Complete.
- NP problems are superset of P problems.
- All the NP problems are non-deterministic in nature.
- Example TSP, Knapsack problem.

Also Read: *Difference Between Prim's And Kruskal's Algorithm*

Difference Between P Problems And NP Problems In Tabular Form

P PROBLEMS	**NP PROBLEMS**
P problems are set of problems which can be solved in polynomial time by deterministic algorithms.	NP problems are the problems which can be solved in non-deterministic polynomial time.

The problem belongs to class P if it's easy to find a solution for the problem.	The problem belongs to NP, if it's easy to check a solution that may have been very tedious to find.
P Problems can be solved and verified in polynomial time.	Solution to NP problems cannot be obtained in polynomial time, but if the solution is given, it can be verified in polynomial time.
P problems are subset of NP problems.	NP problems are superset of P problems.
It is not known whether P=NP. However, many problems are known in NP with the property that if they belong to P, then it can be proved that P=NP.	If P≠NP, there are problems in NP that are neither in P nor in NP-Complete.
All P problems are deterministic in nature.	All the NP problems are non-deterministic in nature.
Selection sort, linear search	TSP, Knapsack problem.

Also Read: *Difference Between NP Hard And NP Complete Problem*

Related Posts:

1. **5 Difference Between NP Hard And NP Complete Problem**
2. **5 Difference Between Deterministic And Non-deterministic Algorithms**
3. **14 Difference Between Digital And Analog Computer**
4. **10 Difference Between Bresenham's And DDA Line Drawing Algorithm**
5. **10 Major Difference Between Flowchart And Algorithm (With Pictures)**
6. **7 Difference Between Prim's And Kruskal's Algorithm With Examples**