P VS NP PROBLEMS

**Main Objective of an Algorithm**

The main objective of an algorithm is to provide a method to solve a problem which has the minimum amount of time complexity and space complexity. Different algorithms are used in a similar problem depending on the average number of inputs and average memory required to run the program. However, many computer researchers are constantly working on methods to reduce the time and space complexity of various algorithms so that many codes implementing these algorithms would be more efficient and fast than before.

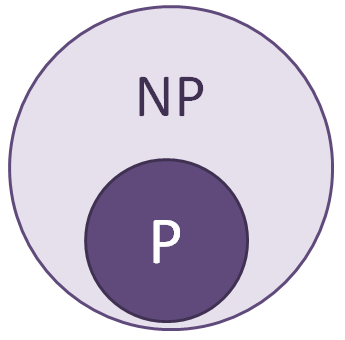
**P Class Problems**

P class problems is a group of various problems which take polynomial time to be executed. Its time complexity is usually O(nk) or log n. These time complexities are acceptable and are pretty fast compared to O(2n) complexity. These problems are also deterministic which means that for the same input, the output will always be the same. Deterministic problems are considered complete or the algorithm has no uncertainties within. Researchers are still trying to make problems within the P Class faster as before.

**NP Class Problems**

NP class problems are a group of problems which are non- deterministic which means that there are uncertainties and gaps within the algorithm. They can be verified in polynomial time. These algorithms usually take problems with time complexity of O(2n) and try to reduce their complexity by adding uncertainties in the algorithm, or generalizing certain parts of the algorithm and leaving empty gaps assuming that it would one day be solved by someone. Once the uncertainties/gaps in the algorithm are solved/removed, the algorithm would be a deterministic algorithm and it would join the P class.

All P class problems were a part of NP class which was later solved and went to the P class. This makes P class a subset of NP class and it is assumed that in the future the NP problems would be in the P class and that P class would be equal to NP class.

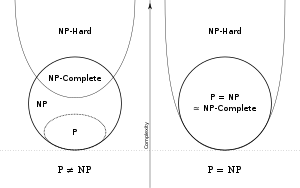


**NP – Hard**

NP-Hard is another class of problems which consist of problems which may or may not be in NP class which can be reduced to a polynomial form. One of the methods to show if the problems can be reduced to a polynomial form is through the CNF-satisfiable method where it breaks every problem into boolean combinations and the total combinations would be 2n showing that the time complexity would be O(2n). If any problem is in the CNF-satisfiable form it is considered an NP-Hard problem.

**NP – Complete**

NP-Complete is another class of problems which are all part of NP-Hard class and part of NP class. These problems can be reduced to CNF-satisfiable form and are also non deterministic to take a polynomial time instead of its deterministic exponential time for execution. These problems are considered satisfied in research work as all the problems in this class could be converted to P class which just the solution of one of the problem.



**Examples of P and NP Problems**

Examples of P Problems:

* Linear Search O(n)
* Binary Search O(log n)
* Insertion/Bubble/Selection Sort O(n2)
* Merge/Quick Sort O(nlog n)
* Matrix Multiplication O(n3)

All these problems have polynomial time of execution and they also have deterministic algorithms. These two things make them part of the P class.

Examples of NP Problems:

* 0/1 Knapsack
* Travelling Salesperson
* Sum of Subsets
* Graph Colouring
* Hamiltonian Cycle

All these problems have their time complexity almost O(2n) and so to make their time complexity in polynomial form, a non-deterministic algorithm is created for these problems thus making them part of the NP class.