

NP-Completeness

I. P

The P stands for *Polynomial-time*. Any algorithm whose time bound is a polynomial or less, is said to run in *Polynomial-time*. Linear search runs in polynomial time because it takes at most n comparisons to determine whether or not an item is in the list. Binary search requires only $O(\lg n)$ comparisons, but is also considered to run in polynomial time. The notation $O(n)$ means on the order of n , while $O(n^2)$ means on the order of n^2 . “On the order of n^2 ” (for example) means that the time bound is a quadratic polynomial of some sort. When talking about time bounds and algorithms we ignore all terms except the biggest and also ignore all coefficients. (The reasons for this are too complicated to discuss here.)

Bubble sort runs in polynomial time because it requires $O(n^2)$ comparisons.

II. N

The N stands for *Nondeterministic*. Nondeterminism in algorithmic theory is a technique for encoding several strongly related computations in a single algorithm. Nondeterministic algorithms are mathematical constructs and cannot be implemented. An algorithm becomes nondeterministic when we introduce a call to the function *select*, as below.

```
A = select(1,2);
```

This statement means that I split the execution into two separate streams. In one stream I assign 1 to A and in the other I assign 2.

In most nondeterministic computations I am not concerned with a final result, but only with whether a computation succeeds or not. A computation succeeds by executing the statement *ACCEPT*. A computation is assumed to succeed if any stream succeeds.

III. Complete

NP-Completeness really isn't about algorithms, it's about problems. The problem of searching an ordered list is a problem that can be solved using linear search or binary search. Sorting is a problem. Bubble sort is not a problem, it is an algorithm.

If an problem can be solved by a polynomial time algorithm it is said to be in the class P. If it can be solved in polynomial time by a non deterministic algorithm, it is said to be in the class NP. P is a subset of NP, but it is not known whether the classes are the same.

A problem is *Complete* for a particular class (such as P or NP), if solving the problem will allow every other problem in the class to be solved as well. There are many NP-Complete problems.