NP-Completeness

P

Consists of classes of problems that are solvable in polynomial time, $O(n^k)$ where n is the size of the input.

NP

Consists of classes of problems that are verifiable in polynomial time using a certificate of a solution.

Problems in P are also in NP, since it can be solved in polynomial time and verified wihtout any certificate. We belive $P \subseteq NP$ but an open question remains if $P \subset NP$.

NPC

A problem is NP-complete if it is in NP and it is as hard as any problem in NP.

Reduction

A is a decision problem we want to solve in polynomial time. We know how to solve a different decision problem B in polynomial time and we have a procedure that transforms the instance α (inputs) af A into some instance β of B such that

- the reduction takes polynomial time
- the answers to the decisions are the same.

The reduction algorithm provides a way to solve A in polynomial time:

1. Use polynomial-time reduction algorithm to transform α of A to β of B
2. Run polynomial-time decision algorithm on β
3. Use answer for β as answer for α

Using polynomial-time reductions, we can show that a problem is NP-Complete and that no polynomial-time algorithm exists for problem B. We have:

$$A \notin P \land T_p(A, B)$$

...