Some Further Complexity Classes – Lecture 18
James Marshall

NP-Complete: the 'hardest' problems in **NP**

Are all the 'hard' problems (in) NP-Complete?

Consider *3-SAT* = $\{\langle w\rangle \mid w$ is a satisfiable 3-CNF Boolean formula$\}$

Design a TM $M(\langle w\rangle)$

$$M(\langle w\rangle) = \begin{cases} \text{accept} & \text{if} \\ \\ \text{loop forever} & \text{otherwise} \end{cases}$$

Definition

NP-Hard: A language A is (in) **NP-Hard** iff

1. All languages in **NP** are polynomial time reducible to A
2. **BUT** A itself need not be in **NP**

NP-Hard: At least as hard as the hardest problems in **NP**

What about space?

Often, computational time can be reduced by increasing computational space, or *vice versa*.

However, space can be re-used… time cannot!

Definitions

Let *f(n)* be a function
Define the *space complexity class* SPACE($f(n)$) to be the collection of all languages decidable by a deterministic Turing Machine in $O(f(n))$ space.

Define the *space complexity class* NSPACE($f(n)$) to be the collection of all languages decidable by a nondeterministic Turing Machine in $O(f(n))$ space

Savitch's Theorem

For any function where

Informally: Although simulating a non-deterministic Turing Machine requires exponential time, it requires space that is polynomial in the size of the input

Definition

$$\text{PSPACE} = \bigcup_k \text{SPACE}\left(n^k\right)$$

Conjecture (what most Computer Scientists believe):

In summary: There is a lot more to learn about computation and complexity