COMP 3721: Theory of Computation

Motivation and Overview

Theory of Computation

> Mathematical study of computing machines, their fundamental capabilities and their limitations.

Q1: What problems are solvable, in PRACTICE, by computer and what problems are not?

Complexity theory (NP-hardness) answers these questions.

Q2: What problems are solvable, in PRINCIPLE, by computers and what problems are not?

Computability theory answers these questions.

Example 1

A compiler can detect syntax errors in the programs you write. Can we write a compiler that will also detect "infinite loops"?

No! The problem of whether a program halts under all inputs is an UNSOLVABLE problem. We will see why in this course.

Example 2

Given any two programs, determine whether they compute the same function.

This problem is also UNSOLVABLE.

Motivation

What is an unsolvable problem? A problem for which there is no algorithm?

What is an algorithm? We need a precise definition of algorithm in order to prove that no algorithm exists for a specific problem.

Model design

These questions lead to the following fundamental tasks:

- Devise a *model of computation* that is as universal as possible – the model should be applicable to existing computers, and it should also correspond to an *abstract notion of computation* that is applicable to any future computation devices.
- Equivalently, devise a *model of computation* that can be applied to programs in any current programming language or in any future programming language.
- We assume that a computer performs one sequential, deterministic computation at a time (not parallel or distributed computers).
- Use this model to characterize problems as solvable or unsolvable.

Some Models

Restricted models of computation (we will discuss all of them)

- Finite automata
- Regular grammars
- Regular expressions
- Pushdown automata
- Context-free grammars

Universal models of computation

- Turing machines (we will only discuss this one)
- Phrase-structure grammars
- Post systems
- Church's lambda calculus
- Markov algorithms

Decision problem – a problem whose output is either *Yes* or *No*.
Example: Given a positive integer n, is it even?

This problem can be transformed into an equivalent *language recognition problem*:

1. Find a way to encode each possible input as a string
2. Let L be the set of all strings corresponding to the 'yes' instances
3. Is the given string in L ?

L is called a *language*. A language is a set of strings over an *alphabet*.