Lecture 005: The Big Ideas

Svadrut Kukunooru

September 23, 2021

In THEORY, all computers, if give enough *time*and *memory*, are capable of computing anything that can be computed.

The A (short for automatic)-Machine, creating by 1940s scientist Alan Turing, is a mathematical model of computation. It, in short, defines a very simple abstract computer.

1. Reads/writes symbols on an infinite memory tape (refer to the theory above)
2. Transitions to a different state based on
    - The current state of progress
    - A finite table of instructions
    - Current symbol on the tape

Every computation can be performed by some Turing Machine.

$$A, B \rightarrow T_{add} \rightarrow A + B.$$

$$A, B \rightarrow T_{mul} \rightarrow A \times B.$$

However, it is important to note that NOT EVERYTHING is computable!

A **UNIVERSAL TURING MACHINE** is a Turing machine that can implement all other turing machines. Its input includes data AND a program.

$$T_{add}, T_{mul}, A, B, C \rightarrow U \rightarrow A(B + C)$$

Most modern machines today are universal Turing machines, such as our phones, laptops, etc. In practice, solving problems with computers involves constraints, such as **time**, **cost**, and **power**.

ABSTRACTION manages complexity by focusing on some attributes over others. Solving a problem using a computer is a systematic sequence of transformations between levels of abstraction.

The sequence is as follows:

PROBLEM STATEMENT

- High-level description in a natural language
- Usually ambiguous, imprecise, or maybe incorrect

ALGORITHMS & DATA STRUCTURES

- Step-by-Step procedures and information organization
- Unambiguous, computable, finite specifications

PROGRAM

- Statements that are organized into modules (e.g. functions, methods, etc.)
- Variables that are designed to contain data organized into arrays, structures, objects, etc.
- High-level, structured, logical

INSTRUCTION SET ARCHITECTURE (ISA)

- Instructions the computer can perform and data types the computer can understand
- Low-level; tied to particular types of machines (e.g. x86, ARM, amd64, etc.)

MICROARCHITECTURE

- Detailed processor implementation (e.g. i3,i5, ryzen 5, etc.)

CIRCUITS

- Logic gates & low-level components

DEVICES

- 100's of millions of electronic switches turning on and off billions of times a second
- Properties of maeterials, manufacturability

Note that there are many options at each level of abstraction; different problems, programming languages, devices, processors, wires, etc.