DSC204A: Scaleable Data Systems Winter 2025

2: Data Representation

Instructor: Umesh Bellur *Scribe: Joshua Chuang, Joyce Hu*

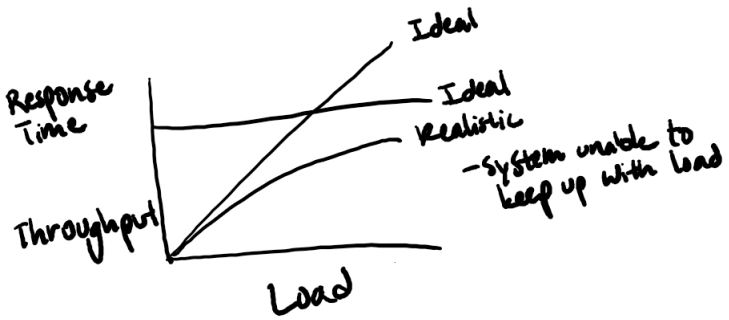
1 Recap

What is a data system?

- retrieves, stores, accesses, and manipulates data
- different components of data system
  - data
  - network
    - retrieving data remotely
  - storage
    - long term storage with disks e.g. solid state disks (SSDs), magnetic disks
    - memory - unlimited memory would be nice but not currently feasible

What is scalability?

- maintaining performance given additional resources or greater load on system (e.g. hundreds of thousands of users)
- what is performance?
  - metrics used to measure and define performance
    - response time - including time of send request over network, time in queue, computation time, time of results sent back
    - throughput - measures requests or work per unit time the system is able to handle
    - throughput related to response time but not the same



- Response time ideally remains constant as load increases
- Throughput (number of requests per second) ideally amount pushed get out in the same amount of time
- Realistically system unable to keep up with load

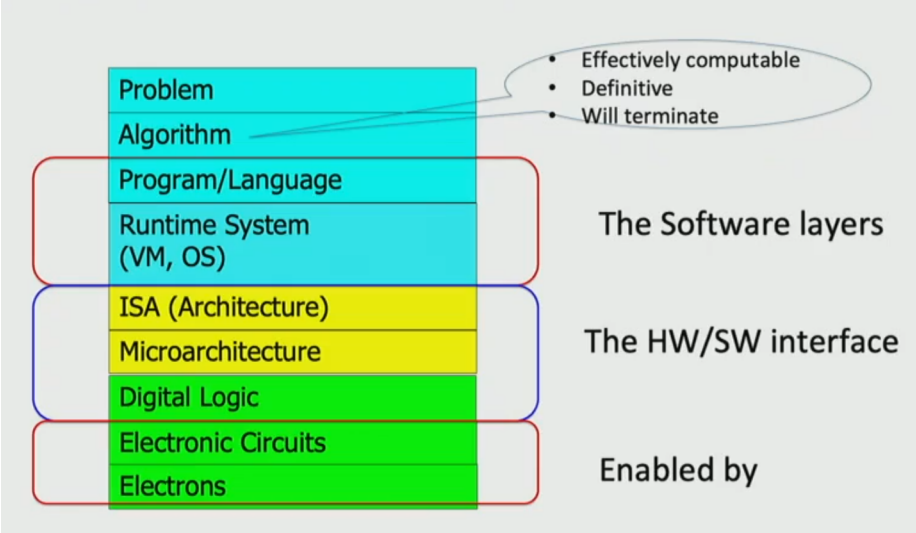
2 Foundation of Data Systems

1. Computer organization (hardware)
   - representation of data
   - electronic machinery (wires, circuits), processors, memory, storage
2. Operating systems
   - layers of abstraction above the hardware
   - instructions, processes, file systems, memory management, data

What is a computer?

- A programmable electronic device that can store, retrieve and process digital data. - Peter Naur

Transformations & Abstractions



- Problem - abstract purpose for system to solve
- Algorithm - abstract description and design of system to solve problem
- Program/Language - translate algorithm to code
- Runtime System (VM, OS)
- ISA (Instruction Set Architecture) - interface exposed by processor for program to use

- Microarchitecture - how processor implements ISA to execute instructions
- Digital Logic - fundamental elements of microarchitecture
- Electronic Circuits/Electrons

3 Computer Organization

3.1 Hardware

- storage, in the form of disks or RAM/memory, is needed to store and retrieve data
- both forms of storage needed because of limitations in cost
- effectively use both by caching
- process data using processors: CPUs, GPUs, TPUs, coprocessors
- networks used to send and retrieve data remotely

Von-Neuman Computer Model - sequential processing

- may have multiple cores and have parallelism across cores
- every core will perform the loop: fetch instruction from memory, execute instruction, increment counter to point to next instruction, and execute the next instruction

3.2 Software

Aspects of Software

- The "mind" of the computer
- instruction - smallest unit of low level commands understood by HW (ISA)
- program (code) - collection of instructions for HW to execute
- programming language (PL) - human-readable formal language to write programs
- application programming interface (API) - set of functions exposed by a program/set of programs for use by humans/other programs
- data - digital representation of information that is stored, processed, displayed, retrieved, or sent by a program

Kinds of Software

- firmware - bios, bootloader of machine
  - load hardware control functions
  - real-only programs

- operating system - manage memory
- application software
  - program or collection of programs to manipulate data (e.g. excel, PostgreSQL)

4 Data Representation

Analog

- machines have issues working with real numbers that can be infinite in number and precision
- analog is the domain of continuous values of finite precision
- errors arise in value representation
  $A + B = C \rightarrow A + \epsilon + B + \epsilon' = C + \epsilon + \epsilon'$
  Errors accumulate with each successive computational step

Digital

- restrict machine to a finite, discrete, set of symbols to reduce potential for error significantly
- shift from analog to digital machines

5 Representation Systems

How did we get to the binary representation of data? An analog machine has issues with precision, so we explore different representations of digital machines.

5.1 Base 1 (Unary)

- simplest way to represent numbers
- ex: number 8 would be represented with 8 repeating symbols
- issue: adding two numbers requires concatenation which takes O(n) time
- overall, the system is simple but not efficient, especially when representing large numbers

5.2 Roman Numeral

- builds upon unary system by using roman numerals to represent numbers
- addresses issue of representing large numbers
- still inefficient to perform operations on numbers

5.3 Base 10 (Decimal)

- Instead of only using one symbol like the unary system, we move to represent numbers using ten symbols (0-9)
- weighted positional notation: the position of each digit represents its value as a power of 10
- allows us to represent decimals, unlike the previous two systems
- large numbers can now be represented more efficiently—O(log n)
- addition can be done in constant time

5.4 Base 2 (Binary)

- a base 10 system works but is held back by limitations of the electrical parts because it requires 10 voltage levels
- it is faster to represent numbers using only 2 voltage levels and is more tolerant of imprecision

6 Digital Representation

All data has to be represented by a sequence of 1s and 0s

- bits: the sequence of 1s and 0s that form a piece of data
- we can apply some layers of abstraction to help us interpret these sequences
- data type: first layer of abstraction to interpret bits (this is determined by the programming language)
  - ex: boolean, integers, floats, strings
- data structure: second layer of abstraction
  - helps us organize instances of different data types into objects with specific properties
  - ex: arrays, linked lists, graphs
- Byte (B): basic unit of data types
  - 1 byte = 8 bits
- Boolean: size is 1B, aka 8 bits
  - actually only needs 1 bit to represent a boolean, so we waste 7 bits
  - this is okay because computers are designed to process 1 byte at a time. it is slower to perform bitwise operations
- Integer: typically 4 bytes but it also depends on the programming language
- Hexadecimal representation: concise representation of binary
  - represent binary using 16 symbols (0-9, A-F)
  - can represent 1 byte of data (8 bits) as 2 hexadecimal symbols instead