CS 4413
Algorithm Analysis

S. Lakshmivarahan
School of Computer Science
University of Oklahoma
USA-73019
Varahan@ou.edu

What is Computer Science (CS)?

- CS is the science of problem solving
- CS is an enabling discipline that helps others in solving their problems

- Key words
  - **Problems**
  - **Solution Process**

Problems

- Arise in various shapes and forms
- Examples include:
  - **Find the roots** of a quadratic equation (where *a*, *b*, c are real numbers) : $ax^2 + bx^2 + c = 0$
  - **Sort a set** of *n* integers
  - **Find a tour** of *n* cities in a map that minimizes the total cost of travel: travelling salesman problem (TSP)

Problems

- **Identify** the mutations of human genome
- **Predict** the motion of a hurricane, the maximum rise in temperature in the next 50 years
- **Compress** a file to minimize storage requirement and compress an image to minimize the transmission time on a link with fixed bandwidth
- **Estimate** the sea surface temp. Distribution near equatorial pacific based on satellite measurements of radiated energy

Problems

- A problem, in general, refers to a collection of **infinite instances** of it
- For example, the problem of finding the roots of $ax^2 + bx^2 + c = 0$ has infinite instances: one each choice of values for *a, b, c*
- **Sort** refers to the problem of sorting a file with *n* keys for each value of the integer *n*
- **Shortest path problem** refers to finding shortest between every pair of nodes in a given graph

Problems

- Every problem has **an intrinsic size** denoted by an integer: *n*
- For **a polynomial**, the degree/number of coefficients
- For **a graph**, number of nodes and edges
- For **a matrix**, the order, $n \times m$
- For **a file**, the number of items to be sorted, searched

Algorithms vs Heuristics

- Pathways for solving a problem are known as
  - Methods
  - Procedures
  - Recipes
  - Algorithms
- We say that a **problem *P*** has **an algorithm *A*** only when *A* can solve all the instances of the problem *P*
- If not, it is known as **a heuristic**

Algorithms vs Heuristics

- Many problems in artificial intelligence are solved by **good heuristics**
- Memory management based on **first-fit/best- fit** are examples of heuristics
- The method of **Gaussian elimination is an algorithm** for solving linear system $Ax = b$

  (A is an n x n nonsingular matrix and b is a n vector.)
- **Dijkstra's algorithm** for finding shortest path between a pair of nodes in a graph

A Classification of Problems

- **Decision Problems**
  - Has yes or no answer
  - Is this file sorted?
  - If not, how to sort?
  - Is the length of the shortest path less than $K(> 0)$?
  - Did the program compile?
- **Optimization Problems**
  - Find the optimal tour in TSP
  - Find a code to maximize compression ratio

Time Measurement

- In CS, we seek algorithms to solve problems of all kinds. The word algorithms is used in a technical context. An algorithm by definition is:
  - A **step-by-step** prescription of actions
  - That is **mechanizable**
  - Takes **finite resources** to solve the given problems
  - Resources are **time** and **space**
  - Space is the amount of **memory** needed
  - Since **memory is cheap**, **time is the only scarce resource**

Time Measurement

- One way to measure time required to solve a problem *P* by an algorithm *A* is by measuring the **wall clock time** taken by the algorithm on a given computer
- A little reflection reveals several difficulties:
  - This wall clock time depends on
    - **Hardware: architecture, technology**
    - **Programing lang.** and **compiler**
    - **Programmer,** etc.

Time Measurement

- We need an **independent framework** for **quantifying time** needed to solve a problem *P* of size *n* by an algorithm *A*
- This is done by choosing **an abstract model of a computers**
- Two models
  - **RAM-model**: Random access memory –we will use it
  - **Turing Machine model**: Sequential access memory- Used in Theory of Computation

RAM Model- Assumptions

- The **RAM model** has
  - Words of **infinite length**
  - **Infinite number** of such words
  - **Store/load** operations are free of cost, i.e., take zero time
    - Perform **basic operations** in one unit of time
      - **Add/subtract**
      - **Multiply/divide**
      - **Compare**
    - That is, RAM is a **unit cost** model.

RAM Model

- We then quantify the time, $T(n)$ required to solve a problem P by an algorithm A as **the work**; measured by the total number of basic operations (+, --, *, %, and ≥) performed by A on a RAM model

  i.e., $T(n)$ = Total work = Total number of basic operations in a RAM model

- If N is the set of all **non-negative integers**, then $T: N \rightarrow N$ (is known as the time **complexity function**)

Comments on RAM Model

1. In practice word **length is finite**: 4 bytes = **32 bits**
2. Thanks to technology with ever decreasing cost of memory, we now can afford very large memory
3. In practice, multiply/divide takes more time compared to add/subtract.
4. In the real machines, it takes considerable **time to load/store** data into memory
5. When you run a program on a real machine, there is **overhead**- load/store- **data movement**

Comments on RAM Model

1. In practice, word length is finite: 4 bytes = 32 bits
   - **Max value of integers** in this word is

     $\pm 2^{31} - 1 = \pm 2{,}147{,}483{,}647$
   - Real numbers are stored as $m_1, m_2 \ldots m_n * 10^{\pm e_1 e_2}$

   If 32 bits are divided into 24 for **mantissa** and 8 for **exponent**, then max value of exponent is $\pm 2^7 - 1 = \pm 127$
   And that for mantissa is $\pm 2^{23} - 1 = \pm 8{,}388{,}608$

   Thus, the mantissa is no more than 6 digits long

Comments on RAM Model

1. In practice word length is finite: 4 bytes = 32 bits
   - Consequently, we have to deal with floating point **truncation errors**
   - Assume we have a machine that can hold 3 digits of mantissa:

     Let $a=0.925$ $\quad a + b = 1.441 = 0.144 \times 10^{+01}$

     $b=0.516$ $\quad$ Error= 0.001 - **Round off error**
   - In the RAM model, infinite word length implies **no round-off errors**

Comments on RAM Model

2. Thanks to technology with ever decreasing cost of memory, we now can afford very large memory

3. In practice, multiply/divide takes large time compared to add/subtract. We can take the **largest of the time** required to multiply, divide, add, subtract and compare **to be the unit** in unit cost model

4. In the real machines, it takes considerable **time to load/store** data into memory

5. When you run a program on a real machine, there is **overhead**-load/ store, keeping counters for loops, evaluating conditions for branching, etc.

Converting T(n) into time

- Consider a machine that takes 10^{-9} sec/operation
- Let $T(n) = 4n^3$. For $n = 10^6, T(n) = 4 * 10^{18}$ operations.
- Time = $4 * 10^{18} * 10^{-9} = 4 * 10^9$ sec
- Number of seconds in a year = 60 x 60 x 24 x 365

  = 31,536,000

  = 31.54 x 10^6 sec.
- Thus, it would take $\frac{4*10^9}{31.54*10^6} = \frac{4000}{31.54} = 126.823$ years

Converting T(n) into time

- <u>Problem</u>: What is the size of the problem that can be solved in 1 hour on a computer that takes 10^{-9} sec/op using an algorithm with time complexity $T(n) = \log n$, $10n$, $2n^2$, $4n^3$, $20n\log n$, 2^n, and $n!$?

Homework: For Your Computer

- Find the basic clock speed
- Time for load/store
- Time for add/subtract/multiply/divide compare
- The power of a machine is expressed as number of floating point operations per-second (denoted as flops) such as 10 megaflops,10 gigaflops, etc. Here is the scale:

Kilo = 10^3	Peta = 10^{15}
Mega = 10^6	Exa = 10^{18}
Giga = 10^9	Zetta = 10^{21}
Terra = 10^{12}	Yotta = 10^{24}

- Find the power of your machine