ICS 103

Data Structures and Algorithms

International Institute of Information Technology
Hyderabad, India

Motivation

- Amount of data that is being handled is getting huge.
- Name some examples in order of scale
  - About 1 – 5 KB
  - About 1 – 5 MB
  - About 1 – 5 GB
  - About 1 – 5 TB
  - About 1 – 5 PB
  - About 1 – 5 EB
- It is believed that every year we produce as many bits of information as is available in the past.

Motivation

- Amount of data that is being handled is getting huge.
- Name some examples in order of scale
  - About 1 – 5 KB
  - About 1 – 5 MB
  - About 1 – 5 GB
  - About 1 – 5 TB
  - About 1 – 5 PB
  - About 1 – 5 EB

So what is the rate of growth of digital data?

- It is believed that every year we produce as many bits of information as is available in the past.

Motivation

- Amount of data that is being handled is getting huge.
- Name some examples in order of scale
  - About 1 – 5 KB : Compiler symbol table
  - About 1 – 5 MB : Source code of a big application
  - About 1 – 5 GB : Telephone directory
  - About 1 – 5 TB : Digital library
  - About 1 – 5 PB : Google maps
  - About 1 – 5 EB : Search engine data
- It is believed that every year we produce as many bits of information as is available in the past.

So what is the rate of growth of digital data?

Motivation

- What do we do with all this data?
- Think of the contacts list in your mobile phone. Typically,
  - look for a person by name and find his/her telephone number.
  - Once in a while, add/remove names to the list.
- How do you search?

Motivation

- What do we do with all this data?
- Think of the contacts list in your mobile phone. Typically,
  - look for a person by name and find his/her telephone number.
  - Once in a while, add/remove names to the list.
- How do you search?
  - Linear search?

Motivation

- What do we do with all this data?
- Think of the contacts list in your mobile phone. Typically,
  - look for a person by name and find his/her telephone number.
  - Once in a while, add/remove names to the list.
- How do you search?
  - Linear search? Almost never.
  - Binary search?

Motivation

- What do we do with all this data?
- Think of the contacts list in your mobile phone. Typically,
  - look for a person by name and find his/her telephone number.
  - Once in a while, add/remove names to the list.
- How do you search?
  - Linear search? Almost never.
  - Binary search? Not really.

Motivation

- What do we do with all this data?
- Think of the contacts list in your mobile phone. Typically,
  - look for a person by name and find his/her telephone number.
  - Once in a while, add/remove names to the list.
- How do you search?
  - Linear search? Almost never.
  - Binary search? Not really.
  - We will know the answer in this course.

Motivation

- Consider an online application such as Google maps.
- Gives you driving directions from say IIIT-H to your home.
- What is the size of this data?*468 Terabytes according to one count.*
- How can it be done?
- How to store the information? In this case, a graph.
- How to quickly find the route given two points A and B?

Motivation

- Need mechanisms to store the data and also to efficiently access data.
- The study of such mechanisms forms the subject matter of Data Structures.
- A fundamental part of any Computer Science curriculum.
  - several practical issues being addressed even today in important conferences.

About this Course

- We will cover several fundamental data structures including:
  - Arrays
  - Stacks and queues
  - Hash tables
- Other pointer based data structures such as
  - lists
  - trees, heaps
- Special data structures such as:
  - Graphs
  - Amortized data structures

Items to Consider

- Will introduce practical motivations to each of the considered data structures.
- Several problem solving sessions to fully understand the implications of using a data structure.
- Emphasis also on correctness and efficiency.
- Elementary analysis
- A basic introduction to parallelism in computing and also parallel programming.
  - Laboratory sessions are therefore very important.

What is NEW this Year?

- Class duration increased to 85 minutes from 55 minutes.
- Co-taught with Dr. Vikram Pudi.
- Term paper/Project as part of the grade.
  - Excellent ones can be extended to summer project, possibly leading to a publication.
  - So, those of you wishing to do research should aim high.
- Plan to include online surveys.

Yet Another Look at the Syllabus

- Syllabus by week
- Basic Data Structures
  - Processing integers (no need for data structures explicitly)
  - Analysis of algorithms
  - The need for data structures
  - The Need for Different access patterns on arrays
  - Limitations of array based data structures
- Intermediate advanced data structure
  - Hashing
  - Trees

Yet Another Look at the Syllabus

- Advanced data structures
  - Data structures for graphs
  - Same as week 10
  - Advanced Topics -- I
  - Advanced Topics -- II
  - Advanced Topics -- III

Other Policies

- Weekly three lecture hours.
- One hour of tutorial.
- Laboratory session every week
  - about 2-3 problems to be solved
  - TAs to assist.
- Several homework assignments
  - About 7, one every two weeks.
  - Each set to have about 6-7 problems
  - Late submission not allowed, unless notified earlier.
- Strictly, no plagiarism
  - Any detected case of plagiarism to be taken seriously.

Other Policies

- Instructor available via office hours.
- Seek an appointment for meeting outside of office hours.
- Email communication is also OK.
- Very important: Seek help early enough.

Other Policies

- Grading scheme
  - Homework – 15%
  - Mid term exam -I – 20 %
  - Mid term exam -2 – 15 %
  - lab exam -1 – 5 %
  - lab exam -2 – 10 %
  - *Term paper/Project – 10 % (Extra credit of 10 % for exceptional work)
  - End term exam – 25 %
- Subject to minor changes.

A Complete Example – Number Systems

- An example to illustrate that data structures are all pervasive.
- We will consider number systems.
- Number systems are a way to represent numbers
  - Using the representation, can do arithmetic on numbers.
  - Ability to count and do arithmetic is a fundamental civilizational trait.
  - Ancient civilizations also practised different number systems with different characteristics.

Number Systems

- A number system is a way to represent numbers.
- Several known number systems in practice even today.
  - Hindu/Decimal/Arabic system
  - Roman system
  - Binary, octal, hexa-decimal.
  - Unary system
  - ...
- A classification
  - positional
  - non-positional

Number Systems

- Hindu/Decimal system
  - Numbers represented using the digits in {0, 1, ,...., 9}.
  - Example: 8,732,937,309
- Roman System
  - Numbers represented using the letters I, V, X, L, C, D, and M.
  - For instance X represents 10, L represents 50.
  - LX stands for 60, VII stands for 7, what is MMXIII?
  - MMMDDDCCCLLLXXXVVVIII – largest numbers without any overlines/subtractions. What is this number?
- Binary system
  - Numbers represented using the digits 0 and 1.
  - 10111 represents 23.

Number Systems

- Positional (aka value based) number systems associate a value to a digit based on the its position.
  - Example: Decimal, binary, ...
- Non-positional do not have such an association.
  - Example: Unary

Operations on Numbers

- Let us consider operations addition and multiplication.
- Hindu/Decimal system
  - Add digit wise
  - Carry of x from digit at position k to position k+1 equivalent to a value of $x.10^{k+1}$, k > 0.
  - Example: Adding 87 to 56 gives 143.
- Unary system
  - Probably, the first thing we learn.
  - To add two numbers x and y, create a number that contains the number of 1's in both x and y.
  - Example: Adding 1111 to 11111 results in 111111111.

Operations on Numbers

- Roman system
  - A bit complicated but possible.
  - Follow the following three steps:
    - Write the numbers side by side.
    - Arrange the letters in decreasing order of value.
    - Simplify.
  - Example: to add 32 and 67:
    - 32 = XXXII, 67 = LXVII.
    - XXXIILXVII
    - LXXXXVIIII – LXLIX – XCIX
    - Simplified as: XCIX

Operations on Numbers

- Rules such as:
  - If there are 4I's, write it as IV.
  - If there are 4X's, write it as XL.
  - Similar rules apply.
- Careful when starting with numbers such as LXIV.
  - Can replace IV with IIII initially.

Operations on Numbers

- Let us now consider multiplication.
- Typically, multiplication is achieved by repeated addition.
- Decimal system
  - Known approach.
- Roman system
  - How to multiply?
  - Much complicated, but is possible.

Multiplication in Roman Numerals

- Easy to imagine the following approach.
  - Multiplication is repeated addition
- Plus, think of a Roman number as the addition of 1000's + 100's + 50's + 10's + 5's + 1's.
- Multiply by each of these, and add as earlier.
- Example: LXII x XXXVII (62 x 37)
  - Multiply each of LXII by II. Meaning, make 2 copies of each symbol in LXII as LLXXIIII
  - Simplify using the rules of addition to CXXIV.
  - Now, multiply each of LXII by V. Start with LLLLLXXXXXIIIIIIIIII, simplify to CCCX.
  - Multiply LXII by XXX. That can be done in two ways. Either multiply by 3 followed by 10, or directly.

Multiplication in Roman Numerals

- Continuing the example,
  - Let us multiply by LXII by 3 as LLLXXXIIIIII and simplified as CLXXXVI.
  - Now, multiply CLXXXVI by 10 as CCCCCCCCCCLLLLLLLLLLXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXVVVVVVVVVVIIIIIIIIII and simplified as MDCCCLX.
  - Add all the constituents as CXXIV + CCCX + MDCCCLX =CDXXXIV + MDCCCLX = MMCCXCIV.
  - What is this number?

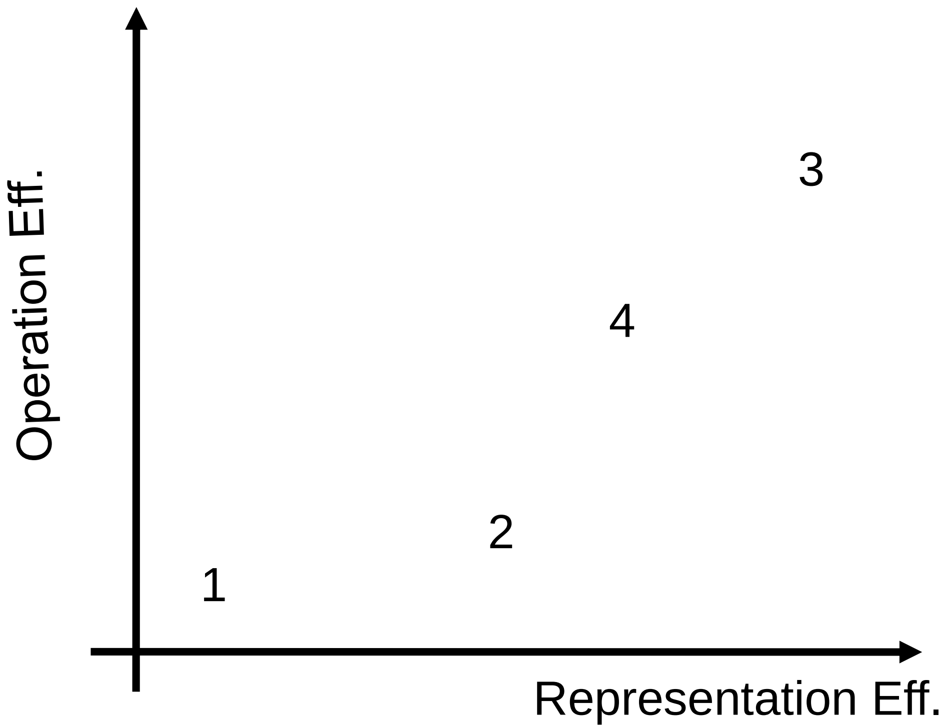
Lesson Learnt

1 – Roman

2 – Unary

3 – Decimal

4 – Binary



- Representation scheme for numbers influences the ease of performing operations.
- Roman system quite difficult to use.
- There are other such systems not in use today.

Are There Other Representation Formats?

- Yes, recall the fundamental theorem of arithmetic.
- Any number can be expressed uniquely as a product of primes.
- So, a product of primes representation is also possible.
- Not easy to add though.

Further Operations

- Let us now fix the decimal system as the representation scheme.
- We will now focus on the efficiency of operations.
- Let us see further operations such as finding the GCD of two numbers.

GCD

- Given two positive numbers, x and y, the largest number that divides both x and y is called the greatest common divisor of x and y. Denoted gcd(x,y).
- Several approaches exist to find the gcd.
- Approach 1 : List all the divisors of both x and y. Find the common divisors, and the largest among the common divisors.
- Example for Approach 1: x = 24, y = 42,
  - divisors of 24 are {1, 2, 3, 4, 6, 8, 12, 24}.
  - divisors of 42 are {1, 2, 3, 6, 7, 14, 21, 42}.
  - Common divisors are {1, 2, 3, 6}. Hence, gcd(24, 42) = 6.

GCD – Approach II

- Use the fundamental theorem of arithmetic and write x and y as:
  - $x = p_1^{a1} . p_2^{a2} .... p_k^{ak}$
  - $y = p_1^{b1} . p_2^{b2} .... p_r^{br}$.
  - It holds that $\gcd(x,y) = p_1^{\min\{a1,b1\}} . p_2^{\min\{a2,b2\}} ... p_r^{\min\{ar,br\}}$.
- Example Approach II, let x = 24, y = 42.
  - $x = 2^3.3$, $y = 2.3.7$.
  - gcd(x,y) = 2.3 = 6.

Which approach is better?

- Both are actually bad from a computational point of view.
- Both require a number to be factorized.
  - a computationally difficult task.
- For fairly large numbers, both approaches require a lot of computation.
- Is there a better approach?
  - Indeed there is, given by the Greek mathematician Euclid.
  - Celebrated as a breakthrough.

Euclid's algorithm for GCD

- Based on the following lemma.
- Lemma : Let x, y be two positive integers. Let q and r be integers such that x = y.q + r. Then, gcd(x,y) = gcd(y, r).
  - Argue that the common divisors of x and y are also common divisors of b and r.
  - Let d divide both x and y. Then, d divides x – yq = r.
  - The converse also applies in a similar fashion.
- The above lemma suggests the following algorithms for gcd.
  - Apply the above lemma repeatedly till the remainder is 0.
  - Let r1, r2, ..., be the remainders.

Euclid's Algorithm for GCD

- Let r2, r3, ..., be the remainders with r0 = x and r1 = y.
- We have that:

  r0 = r1q1 + r2,

  r1 = r2q2 + r3

  r2 = r2q3 + r4

  and so on, till

  r_{n-1} = rn qn + 0
- By the result of the above lemma, it also holds that:

  gcd(r0, r1) = gcd(r1, r2)

  = gcd(r2, r3)

  = ...

  = gcd(r_{n-1}, rn)

  = gcd(rn, 0) = rn
- Notice that rn is the last nonzero remainder in the process.

Euclid's Algorithm

```
Algorithm GCD-Euclid(a,b)
    x := a, y := b;
    while (y \ne 0)
        r := x mod y;x := y;y := r;
    end-while
End-Algorithm.
```

- Example, x = 42 and y = 24.
- Iteration 1: r = 18, x = 24; y = 18
- Iteration 2: r = 6, x = 18, y = 6
- Iteration 3: r = 0.

Euclid's Algorithm

- Why is this efficient?
- It can be shown that given numbers x and y, the algorithm requires only about log min{x,y} iterations.
  - Compared to about sqrt{x} for Approach I.
  - Why does approach 1 takes sqrt{x} iterations?
- There is indeed a difference for large numbers.
- The example suggests that also efficient ways to perform operations are of interest.

Modular Arithmetic

- Again, integer operations with several applications.
  - E.g,. cryptography
- Let m be a positive integer. Then, for two positive integers a and b, we say that a = b (mod m) iff there exists an integer k such that a = b + km.
  - Example: 12 = 2 mod 5, 37 = 1 mod 4.
- With the above definition, we can write
  - If a = b mod m, and c = d mod m, then a+c = (b+d) mod m, and ac = bd (mod m).
  - Modular addition: (a+b) mod m = ((a mod m) + (b mod m)) mod m.
    - Example: a = 2 mod 5, b = 4 mod 5, then a+b = 1 mod 5.
  - Similarly, modular multiplication: ab (mod m) = ((a mod m).(b mod m)) mod m.
    - Example: With the above a, b, ab = 3 mod 5.

Modular Multiplication

- An important operation in cryptography.
- Given integers b, n, and m, compute b^n (mod m).
- One can simply compute b^n and then take the modulo wrt m.
- But becomes impractical for even moderate values of b and n.
- One improvement : Recall the definition of modular multiplication.
  - Can take b^k mod m whenever b^k exceeds m.
  - Continue with b^k mod m.
  - Example:
    - $b = 5$, $n = 4$, $m = 6$. $b^2 = 25 = 1 \bmod 6$.
    - Now, $b^3 = 5 \bmod 6$, and $b^4 = 1 \bmod 6$.

Modular Multiplication

- A further improvement is possible with the following approach.
- Let n be written in binary as $n_{r-1}n_{r-2}...n_1n_0$.
- Then, the exponent of b can be rewritten as
  - $n_{r-1}2^{r-1}+n_{r-2}2^{r-2}+...+2^1n_1+2^0n_0$
  - Now, if for any i between 0 and r-1, if any $n_i = 0$, then $n_i2^i = 0$, and $b^0 = 1 \pmod{m}$.
  - Such indices have no effect on b^n mod m.
  - Thus, we need to evaluate only indices where $n_i = 1$.

Modular Multiplication

- Example, let b = 4, n = 9, and m = 11.
- $n = (1001)_2$
- So, we need to evaluate 4^8 (mod 11) and 4^1 (mod 11).
- How to compute the former?
- Realise that for $k = 2^r$, b^k (mod m) can be computed by computing b^2 mod m, $b^4 = b^2.b^2$ mod m, and so on for r iterations.
- Putting together everything, the improved modular exponentiation algorithm follows.

Algorithm for Modular Exponentiation

```
Algorithm ModExp(b, n, m)
Begin
    Let n = (n_{r-1} n_{r-2} ... n_1 n_0)_2
    result = 1;
    power =  a mod n;
    for i=1 to r-1 do
     if ni = 1 then
         result = (result . Power) mod m
      power = power . Power
   End
End
```

About the Algorithm

- Not very difficult to execute the if condition. Why?
- Try a few examples offline.
- There are log n iterations of the for loop.
- Each iteration requires at most two multiplications and two modulo operations.
- Each multiplication, and also modulus, cab be computed in about log m bit operations.

More on Integers

- Presently, some computations require us to work with numbers that are more than 200 digits.
  - Example: RSA cryptography.
- How to process such large numbers with a computer?
  - A problem of huge practical interest.
  - Few solutions, but scope for improvement is still there.
  - A current research area for some.

Laboratory Session

- Problem 1: Implement routines to add and multiply two Roman numbers.
- Problem 2: Implement Euclid's Gcd algorithm.
- Problem 3: Implement the routines to do modular exponentiation.

Acknowledgements

- To several online sources about the Roman number system.
- To Pranav for initiating a discussion on number systems in one meeting.