UMBC

AN HONORS UNIVERSITY IN MARYLAND

IS 709/809: Computational Methods for IS Research

Math Review: Algorithm Analysis

Nirmalya Roy
Department of Information Systems
University of Maryland Baltimore County

Why do we need math in Algorithm Analysis?

- Analyzing data structures and algorithms
  - Deriving formulae for time and memory requirements
  - Will the solution scale?
- Proving algorithm correctness
  - similar to proving a mathematical theorem; fundamentally, it is algorithm-dependent
  - to prove the incorrectness of an algorithm, one counter-example is enough

Floors and Ceilings

- Floor operation
  - Denoted: floor(x) or $\lfloor x \rfloor$
  - Greatest integer less than or equal to x
  - E.g. floor(5.4) = ?, floor(5.9) = ?
  - E.g. floor(-5.4) = ?, floor(-5.9) = ?
- Ceiling operation
  - Denoted: ceiling(x) or $\lceil x \rceil$
  - Smallest integer greater than or equal to x
  - E.g. ceiling(5.4) = ?, ceiling(5.9) = ?
  - E.g. ceiling(-5.4) = ?, ceiling(-5.9) = ?

Floors and Ceilings (cont'd)

- *floor*(x), denoted $\lfloor x \rfloor$, is the greatest integer ≤ x
- *ceiling*(x), denoted $\lceil x \rceil$, is the smallest integer ≥ x
- Normally used to divide input into integral parts

$$\left\lfloor \frac{N}{2} \right\rfloor + \left\lceil \frac{N}{2} \right\rceil = N$$

Exponents

- Written x^a, involving two numbers, x and a
  - x is the base
  - a is the exponent
- If a is a positive integer
  - $x^a = x \bullet x \bullet \ldots \bullet x$ (a times)
- x^n read as
  - “x raised to the n-th power”
  - “x raised to the power n”
  - “x raised to the exponent n”
  - “x to the n”

Exponents (cont'd)

- $x^0 = 1, x \neq 0$
- $x^{-n} = 1/x^n, x \neq 0$
- $x^a \bullet x^b = x^{(a+b)}$
- $x^a / x^b = x^{(a-b)}, x \neq 0$
- $(x^a)^b = x^{ab}$
- $(xy)^a = x^a \bullet y^a$
- $x^n + x^n = 2x^n \neq x^{2n}$
- $2^n + 2^n = 2^{n+1}$

Logarithm

- Definition
  - $x^a = b$ if and only if $\log_x b = a$
  - $\log_x b$ read as "logarithm of b to the base x"
- The power or exponent to which the base x must be raised in order to produce b
- E.g. $\log_{10} 1000 = 3$
- E.g. $\log_2 32 = 5$
- Only positive real numbers have real number logarithms

Logarithm (cont'd)

- Rules of logarithms
  - $\log_a b = \log_c b / \log_c a$,   s.t. $a, b, c > 0, a \neq 1$
  - Proof: will be derived in the class
  - Useful for computing the logarithm of a number to an arbitrary base using the calculator
  - In computer science, $\log a = \log_2 a$ (unless specified otherwise)

Logarithm (cont'd)

- Rules of logarithms
  - $\log(ab) = \log a + \log b, \quad a, b > 0$
    - Proof: will be derived in the class
  - $\log(a/b) = \log a - \log b$
  - $\log(a^b) = b \log a$
  - $\log x < x$ for all $x > 0$
  - $\log 1 = 0$
  - $\log 2 = 1$
  - $\log 1{,}024 = 10$
  - $\log 1{,}048{,}576 = 20$
  - $\lg a = \log_2 a$

$\ln a = \log_e a$
where $e = 2.7182...$

ln: natural logarithm

- How many times to halve an array of length n until its length is 1?

Factorials

- Denoted: $n!$
- Read: “n factorial”
- Definition:
  - $n! = 1$ if $n = 0$
  - $= n\,(n - 1)!$ If $n > 0$
- $n! < n^n$
- How many different ways of arranging n distinct object into a sequence (called permutation of those objects)? $n!$

Series

- General $\sum_{i=0}^{N} f(i) = f(0) + f(1) + \ldots + f(N)$

- Linearity $$\sum_{i=0}^{N} [f(i) + g(i)] = \sum_{i=0}^{N} f(i) + \sum_{i=0}^{N} g(i)$$

$$\sum_{i=0}^{N} (cf(i) + g(i)) = c\sum_{i=0}^{N} f(i) + \sum_{i=0}^{N} g(i)$$

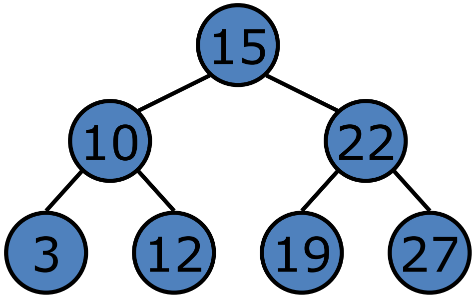
Arithmetic Series

$$\sum_{i=1}^{N} i = \frac{N(N+1)}{2}$$

$$\sum_{i=1}^{N} i^2 = \frac{N(N+1)(2N+1)}{6}$$

$$\sum_{i=1}^{N} c = cN$$

- How many nodes are there in a complete binary tree of depth D?



Geometric Series

- Geometric series

$$\sum_{i=0}^{N} A^i = \frac{A^{N+1}-1}{A-1}$$

$$\sum_{i=0}^{N} 2^i = 2^{N+1}-1$$

$$\sum_{i=0}^{N} A^i \le \sum_{i=0}^{\infty} A^i = \frac{1}{1-A}; \quad \text{if } 0 < A < 1$$

$$\text{As N} \to \infty, \sum_{i=0}^{N} A^i \to \frac{1}{1-A}, \quad \text{if } 0 < \text{A} < 1$$

- Proof: will be derived in the class

- Example: Compute $\sum_{i=0}^{\infty} \frac{i}{2^i}$

Modular Arithmetic

- A is congruent to B modulo N, written as
  $A \equiv B \pmod{N}$
  if N divides (A – B).
- This means that the remainder is the same when either A or B is divided by N.
- (A mod N) = (B mod N) => $A \equiv B \pmod{N}$
  - E.g., $81 \equiv 61 \equiv 1 \pmod{10}$
- Note: A mod N = A – N * $\lfloor A / N \rfloor$

This is the remainder

Modular Arithmetic (cont'd)

- Example:
  - $104 \equiv 79 \equiv 4 \pmod{25}$
  - $33 \equiv 3 \pmod{10}$

- If $A \equiv B \pmod{N}$

  Then $(A + C) \equiv (B + C) \pmod{N}$

  and $AD \equiv BD \pmod{N}$

- Application: Basis of most encryption schemes: (Message mod Key)

Summary Math Review

- Proof Techniques
  - Proof by induction
  - Proof by contradiction
  - Proof by counterexample
- Recursion
- Exponents, logarithm, arithmetic series, geometric series, modular arithmetic etc

Questions

?