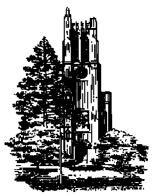
Wednesday June 8, 2016
Lecture 14

Algorithms Complexity



Notables

- Tentative Schedule for the week

Week	M	T	W	R	Topic	Section
1	6-6				Countable and uncountable sets	2.5
		6-7			Algorithms	3.1
			6-8		More on Algorithms	3.1
				6-9	Growth functions	3.2

Homework#8, Due Thursday (6/10)

- Sort the list [64, 13, 29, 32, 21, 46, 60] using
  - Exchange sort; show each pass
  - Bubble sort; show each pass
  - Merge sort; show each sublist
  - For each case, give the number of 2-number comparisons required.
- Use Newton's method to find the square root of 27 up to 2 decimal places. Use 7 as your initial guess.
- Page 204, Problem 52
- Page 216, Problem 2, 8, and 14
- Page 217, 34(a)
- Page 218, 74(a), and 74(b)

Complexity of An Algorithm

- The *complexity* of an algorithm is the cost, measured in some parameter (such as the running time, amount of storage, number of particular operations, etc) of using the algorithm.
- To express the complexity, we need to
  - Decide on the cost parameter(s).
  - Describe the cost function. How?

The Impact of Data Representation

- Consider the following tasks
  - T1: Adding two numbers.
  - T2: Deciding whether a number is even.
- Now, consider the following ways of representing numbers. Take, for example, integer twelve.
  - R1: "usual" representation: Example: 12
  - R2: "long" representation: Example: 111111111111
- Which representation is "preferred" for which task?

Testing if n is prime

1. Input: A positive integer $n > 1$.
2. For $i = 2$ to $n^{1/2}$ Do
   (a) Divide n by i, and let r be the remainder.
   (b) If $r = 0$ output "n is not prime", stop.
   End For
3. Output "n is prime", stop.

Worst-case analysis: cost = $d \cdot n^{1/2}$ where d is some constant.

Primality Testing …..

- So, if $n = 113$, the cost would be about $d \cdot n^{1/2} = d \cdot (113)^{1/2} = 11d$.
- How do we represent number n?
  - Using R1?
  - Using R2?
- The convention is to express the cost function in terms of the "minimum amount of information" (referred to as the input size) needed to describe the algorithm.

423

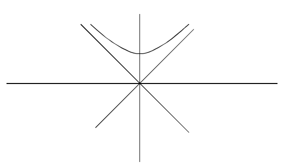
The Growth Functions

- The goal is to be able to make quantitative assertions about algorithms for the purpose of comparing the "goodness" of algorithms, when the input size is "VERY LARGE".
- The Big-*Oh* notation, $O(\,)$
- The Big-Omega notation, $\Omega(\,)$
- The Big- Theta notation, $\Theta(\,)$

424

The Big-*Oh* notation, *O*( )

- Developed by Backmann, 1894, for asymptotic analysis
- The word "asymptotic" stems from a Greek root meaning "not falling together".



425

Facts about log function

Logarithms and Log Properties

Definition

$y = \log_b x$ is equivalent to $x = b^y$

Example

$\log_5 125 = 3$ because $5^3 = 125$

Special Logarithms

$\ln x = \log_e x$ natural log

$\log x = \log_{10} x$ common log

where $e = 2.718281828\ldots$

Logarithm Properties

$\log_b b = 1$ $\quad \log_b 1 = 0$

$\log_b b^x = x$ $\quad b^{\log_b x} = x$

$\log_b \left(x^r\right) = r \log_b x$

$\log_b (xy) = \log_b x + \log_b y$

$\log_b \left(\frac{x}{y}\right) = \log_b x - \log_b y$

The domain of $\log_b x$ is $x > 0$

426

The Big-*Oh* notation, *O*( )

- The notation allows us to suppress unimportant detail and concentrate on the salient features; we can hide the details without losing the big picture
- Definition: A *positive* function $f(x)$ is Big O of another *positive* function $g(x)$, written as $f(x) = O(g(x))$, and read as $f(x)$ is Big-*Oh* of $g(x)$, if there are constants c and k such that

  $f(x) \le c\, g(x)$ for all $x > k$

427

The Big-*Oh* notation, *O*( )

- By saying that $f(x)$ is Big-*Oh* of $g(x)$, we mean that f certainly does not "grow" at a faster rate than g; It might grow at the same rate, or it might grow more slowly.

428

Example

Show that $f(x) = \frac{x^3}{3} + \frac{x^2}{2} + \frac{x}{6}$ is $O(x^3)$.

Solution: $\frac{x^3}{3} + \frac{x^2}{2} + \frac{x}{6} \le \frac{x^3}{3} + \frac{x^3}{2} + \frac{x^3}{6} = x^3$ for $x \ge 0$

Thus, $f(x) \le cx^3$ when $c \ge 1$ $x \ge 0$

Note that we could also be sloppy and write $f(x) = O(x^{10})$. Nothing in the definition of O requires us to give a best possible bound.

429

Example

Show that $\log n = O\left(\sqrt[m]{n}\right)$ $\forall m \in Z^+$

Solution:

430

Example

Show that $\log n = O\left(\sqrt[m]{n}\right)$ $\forall m \in Z^+$

Solution:

First note that $n \le 2^{n-1}$ $\forall n \in Z^+$

Thus, $\log_2 n \le n - 1 \Rightarrow \log n < n$.

Now, $\log n = \frac{m}{m} \log n = m \log n^{1/m} = m \log \sqrt[m]{n} < m\sqrt[m]{n}$

$\Rightarrow \log n = O\left(\sqrt[m]{n}\right)$

431

Example

- Determine whether each of these functions is $O(x)$. For each possible case, give a c and a k.
  1. $f(x) = 10$
  2. $f(x) = 3x + 7$
  3. $f(x) = x^2 + x + 1$
  4. $f(x) = 5\log x$
  5. $f(x) = \lfloor x \rfloor$
  6. $f(x) = \lceil x/2 \rceil$

432

Example

- Determine whether each of these functions is $O(x)$. For each possible case, give a c and a k.
  1. $f(x) = 10$
     - $f(x) \le cx$, $c = 1$ an $k = 10$, or $c = 0.5$ and $k = 20$, etc
  2. $f(x) = 3x+7$
     - $f(x) = 3x+7 \le cx$, $c = 4$, $k = 7$, or $c = 5$ and $k = 4$, etc
  3. $f(x) = x^2 + x + 1$
     - No such c and k exist
  4. $f(x) = 5\log x$
     - $f(x) = 5\log x \le cx$, $c = 5$ an $k = 1$, etc
  5. $f(x) = \lfloor x \rfloor$
     - $f(x) = \lfloor x \rfloor \le cx$, $c = 1$ an $k = 0$, etc
  6. $f(x) = \lceil x/2 \rceil$
     - $f(x) = \lceil x/2 \rceil \le cx$, $c = 1$ an $k = 0$, etc

433

Example

- Find the least integer n such that $f(x)$ is $O(x^n)$ for each of these functions.
  1. $f(x) = 2x^3 + x^2 \log x$
  2. $f(x) = 3x^3 + (\log x)^4$
  3. $f(x) = (x^4 + x^2 + 1)/(x^3 + 1)$
  4. $f(x) = (x^4 + 5 \log x)/(x^4 + 1)$

434

Example

- Find the least integer n such that $f(x)$ is $O(x^n)$ for each of these functions.
  1. $f(x) = 2x^3 + x^2 \log x$
     - $O(x^3)$
  2. $f(x) = 3x^3 + (\log x)^4$
     - $O(x^3)$
  3. $f(x) = (x^4 + x^2 + 1)/(x^3 + 1)$
     - $O(x^1)$
  4. $f(x) = (x^4 + 5 \log x)/(x^4 + 1)$
     - $O(x^0)$
       - Use $c = 2$, and $k = 1$ (for example)

435

Example

Show that $n^m = O(2^n) \quad \forall m \in Z^+$

For example, taking $m = 1000$,

we would have $n^{1000} = O(2^n)$

Solution hint:

Knowing that $\log n < n$ could be helpful.

436

Example

Show that $n^m = O(2^n) \quad \forall m \in Z^+$

Solution:

Remember that $\log n < n$.

$\log_2 n^m = m \log_2 n = 2m \log \sqrt{n} < 2m\sqrt{n}$

$\log_2 n^m < 2m\sqrt{n} = \dfrac{2m \cdot n}{\sqrt{n}}$ But $\dfrac{2m}{\sqrt{n}} \le 1 \quad \forall n \ge 4m^2$

Therefore, $\log_2 n^m < n \qquad \forall n \ge 4m^2$

$\Rightarrow \qquad n^m < 2^n \qquad \forall n \ge 4m^2$

$\Rightarrow \qquad n^m = O(2^n)$

437

Big-*Oh* Hierarchy

$1, \log n, \ldots, \sqrt[4]{n}, \sqrt[3]{n}, \sqrt[2]{n}, \sqrt{n}, n,$

$n \log n, n\sqrt{n}, n^2, n^3, n^4, \ldots, 2^n, n!, n^n$

Each term is the Big - *Oh* of any term to the right of it.

438

The Big-Omega Notation, Ω( )

- Definition: A positive function $f(x)$ is Big-Omega of another positive function $g(x)$, written as $f(x) = \Omega(g(x))$, if and only if
  $g(x) = O(f(x))$.
- Ω-notation is used to express "lower bound". For example, finding a maximum element in a list of n elements would require $\Omega(n)$ two-number comparisons.

439

The Big-Omega notation, Ω( )…

- An algorithm whose running time is $\Omega(n^2)$ is inefficient compared with another algorithm whose running time is $O(n\log n)$.

440

The Big-Theta Notation, $\Theta(\,)$

- Definition: A positive function $f(x)$ is Big Θ of another positive function $g(x)$, written as $f(x) = \Theta(g(x))$, if there are constants c_1, c_2, and k such that

  $c_1\, g(x) \le f(x) \le c_2\, g(x)$ for all $x > k$
- Equivalently,

  $f(x) = \Theta(g(x)) \Leftrightarrow f(x) = O(g(x)) \wedge f(x) = \Omega(g(x))$
- Equivalently,

  $f(x) = \Theta(g(x)) \Leftrightarrow f(x) = O(g(x)) \wedge g(x) = O(f(x))$
- Example: $(x + 1)^2 = \Theta(3x^2)$

O, Ω, Θ, Recap, interpretation

- $f(x) = O(g(x))$
  - Function f grows either slower than function g or at the same rate as function g, but not faster than g.
- $f(x) = \Omega(g(x))$
  - Function f grows at least as fast as g; it can even grow faster.
- $f(x) = \Theta(g(x))$
  - Functions f and g grow at the same rate.

Using Limit to Express O, Ω, Θ

$$\text{If } \lim \frac{f(x)}{g(x)} = 0 \quad \Rightarrow \quad f(x) = O(g(x))$$
$$g(x) \ne O(f(x))$$
$$f(x) \ne \Theta(g(x))$$

$$\text{If } \lim \frac{f(x)}{g(x)} \in R^{+} \quad \Rightarrow \quad f(x) = O(g(x))$$
$$g(x) = O(f(x))$$
$$f(x) = \Theta(g(x))$$

$$\text{If } \lim \frac{f(x)}{g(x)} = +\infty \quad \Rightarrow \quad f(x) \ne O(g(x))$$
$$g(x) = O(f(x))$$
$$f(x) = \Omega(g(x))$$