Assignment 01

Paul Jones and Matthew Klein

Professor Professor Kostas Bekris

Design and Analysis of Computer Algorithms (01.198.344)

February 5, 2014

Part A

Problem 1

In each of the following situations indicate whether $f = O(g)$ or $f = \Omega(g)$ or $f = \Theta(g)$:

1. $f(n) = \sqrt{2^{7x}}, g(n) = \lg(7^{2x})$

$$f(n) = \sqrt{2^{7x}} = \sqrt{128^x}$$

$$g(n) = \lg(7^{2x}) = \lg(49^x)$$

$$lg(49^1) \approx 5.6$$

$$\sqrt{128^1} \approx 11.3$$

   Notice that both of these functions only grow relative to x.

$$f = \Omega(g)$$

2. $f(n) = 2^{nln(n)}, g(n) = n!$

   The factorial, that is $n!$, function grows much, much faster than 2^n.

$$f = \Omega(g)$$

3. $f(n) = \lg(\lg^*(n)), g(n) = \lg^*(\lg(n))$

$$f = \Theta(g)$$

4. $f(n) = \frac{lg(n^2)}{n}, g(n) = lg^*(n)$

$$f(n) = \frac{lg(n^2)}{n} = \frac{2lg(n)}{n}$$

$$f = \Theta(g)$$

5. $f(n) = 2^n, g(n) = n^{\lg(n)}$

   This is comparing the exponential function to a function that is less than n^2.

$$f = \Omega(g)$$

6. $f(n) = 2^{\sqrt{\ln(n)}}, g(n) = n(\lg(n)^3)$

$$f(n) = 2^{\sqrt{n}}, g(n) = (2^n)(n^3)$$

$$f = \Omega g$$

7. $f(n) = e^{\cos(x)}, g(n) = \lg(x)$

$$f = \Omega(g)$$

8. $f(n) = \lg(n^2), g(n) = (\lg(n))^2$

$$f = \Theta(g)$$

9. $f(n) = \sqrt{4n^2 - 12n + 9}, g(n) = n^{\frac{3}{2}}$

$$f = \Theta(g)$$

10. $f(n) = \sum_{k=1}^{n} k, g(n) = (n+2)^2$

$$f = \Omega(g)$$

Problem 2

Part B

Problem 3

Problem 4

Problem 5

Part C

Problem 6

Problem 7

Part D

Problem 8

Problem 9