Seoul National University

M1522.000900 Data Structure

Homework 2: Algorithm Analysis (Chapter 3)

Computer Science & Engineering

2017-18538 Hwang Sun Young

Question 1

$n!$ → $3 \cdot 2^n$ → $6n^2$ → $20\,n$ → $log_2 n$ → $log_2\, log_2 n$ → 2^{10}

2^n is in O($n!$), n^2 is in O(2^n), n is in O(n^2), $log \ \ n$ is in O(n).

Because $log \ \ n$ is in O(n), $log \ \ log \ \ n$ is in O($log \ \ n$). 1 is in O($log \ \ log \ \ n$)

Question 2

(1) f(n) = log n^2 and g(n) = log n + 5

f(n) = log n^2 = 2 log n

(a) f(n) = O(g(n)) since 2 log $n < 2$ (log $\mathrm{n} + 5$)

(b) f(n) = Ω(g(n)) since 2 log $n \geq$ log $n + 5$ for all n > n_0 = 32 (n_0 is positive constant)

(c) Because big-Oh and Ω coincide, f(n) = **Θ**(g(n))

(2) f(n) = $\sqrt{n}$ and g(n) = log n^2

f(n) = $\sqrt{n}$ = $\sqrt[4]{n^{\frac{1}{2}}} \geq \frac{1}{4}$ log n^2

c and n_0 are positive constants.

(a) f(n) ≠ O(g(n)) since $\sqrt{n}$ ≥ log n^c for any c and any n > n_0.

(b) f(n) = Ω(g(n)) since $\sqrt{n}$ = $\sqrt[4]{n^{\frac{1}{2}}} \geq \frac{1}{4}$ log n^2

(c) Because f(n) = Ω(g(n)) but f(n) ≠ O(g(n)), f(n) ≠ **Θ**(g(n)).

(3) f(n) = n and g(n) = $log^2 n$

c and n_0 are positive constants.

g(n) = $log^2 n$ = 2 log n

$\sqrt{n}$ ≥ log $\sqrt{n}$ = $\frac{1}{2}$log n

n ≥ $\frac{1}{4}log^2 n$

(a) f(n) ≠ O(g(n)) since n ≥ $c\,log\ n$ for any c and any n > n_0.

(b) f(n) = Ω(g(n)) since n ≥ $\frac{1}{4}\ log^2 n.$

(c) Because f(n) = Ω(g(n)) but f(n) ≠ O(g(n)), f(n) ≠ **Θ**(g(n)).

(4) f(n) = log n^2 and g(n) = $log^2 n$

c and n_0 are positive constants

f(n) = log n^2 = 2 log n, g(n) = $log^2 n$ = $(\log n)^2$

(a) f(n) = O(g(n)) since log n ≤ c$(\log n)^2$ for any c and any n > n_0.

(b) f(n) ≠ Ω(g(n)) since log n ≤ c$(\log n)^2$ for any c and any n > n_0.

(c) Because f(n) = O(g(n)) and f(n) ≠ Ω(g(n)), f(n) ≠ **Θ**(g(n)).

Question 3

(1) 1st for-loop is in **Θ**(N), 2nd for-loop is in **Θ**(M) for time complexity. So the time complexity of following code is **Θ**(N+M). In terms of space complexity, **Θ**(1). Because, a occupy 1, b occupy 1, i and j occupies 1. (total 3)

(2) this code is double for-loop. Outer loop is in **Θ**(n) and inner loop is in **Θ**(log n). So the time complexity of the following code is **Θ**(n log n).

(3) (b) X will always be a better choice for large inputs.

Question 4

(1) $T(0) = c$

$T(n) = 3T(n-1) + 2$

$= 3(3T(n-2) + 2) + 2$

$= 3^2T(n-2) + 3 \cdot 2 + 2$

$= 3^2(3T(n-3) + 2) + 3 \cdot 2 + 2$

$= 3^3T(n-3) + 3^3 2 + 3 \cdot 2 + 2$

Then, $T(n) = 3^n T(n-n) + (3^n + 3^{n-1} + \cdots + 3 + 1) \cdot 2$

$= 3^n T(n-n) + 3^n - 1$

$= 3^n (T(0) + 1) - 1$

$= 3^n (c + 1) - 1$

Then, T(n) is in **Θ (3^n).**

(2)

n: size of input that can be processed in t second in machine X

N: size of input that can be processed in t second in machine Y

f(s): number of operations for input of size s

$f(N) = 27f(n) = 3^3 f(n) = 3^{n+3}$ $(f(n) = 3^n)$

machine Y takes t seconds for n+3 inputs.

Question 5

(1) $T(n) = T(n-1) + 1$, $T(0) = 1$.

$T(n) = T(n-1) + 1 = T(n-2) + 2 = \cdots = T(n-n) + n = n + 1$

T(n) is in O(n) since $T(n) = n + 1 < 2n$.

T(n) is in Ω (n) since $T(n) = n + 1 > n$.

Then T(n) is in **Θ**(n).

(2) $T(n) = T(\frac{n}{2}) + 3$, $T(0) = 1$.

According to the Figure 2, T(1) = T(0) = 1.

$T(n) = T(\frac{n}{2}) + 3 = T(\frac{n}{2^2}) + 2*3 = \cdots = T(\frac{n}{2^{logn}}) + (\log n)*3 = T(1) + 3 \log n = 3 \log n + 1$

T(n) is in O(log n) since $T(n) = 3 \log n + 1 < 4 \log n$.

T(n) is in Ω (log n) since $T(n) = 3 \log n + 1 > 3 \log n$.

Then T(n) is in **Θ**(log n).