GROUP: Mamo-Sal Sort
Benzon Carlitos Salazar
Abraham Mamo
CS433 - Theory of Algorithms

Homework 1

1. Growth Of Functions
   a. $3n^3 + 19n^2 + 10n + 1700 = \Omega(n^3)$

   Proof. Let $f(n) = 3n^3 + 19n^2 + 10n + 1700$, $g(n) = n^3$, $c = 1700$, $n_0 = 1$.
   $\therefore\ c * g(n) = 1700 * n^3 < 3n^3 + 19n^2 + 10n + 1700 = f(n),\ n \geq n_0$.

   □

   b. $13n^2 + 6n + 150 = O(n^2)$

   Proof. Let $f(n) = 13n^2 + 6n + 150$, $g(n) = n^2$, $c = 170, n_0 = 1$.
   $\therefore\ c * g(n) = 170 * 1^2 \geq 13(1)^2 + 6(1) + 150 = f(n),\ n \geq n_0$.

   □

   c. If $f_1(n) = O(g_1(n))$ and $f_2(n) = O(g_2(n))$, then $f_1(n) * f_2(n) = O(g_1(n) * g_2(n))$, where all the functions are positive.

   Proof. Let $f_1(n) \leq c_1 * g_1(n_1), f_2(n) \leq c_2 * g_2(n_2)$, and $N_i = max(n_1, n_2)$.

   $$\begin{aligned} \therefore\ f_1(n) * f_2(n) &\leq (c_1 g_1(n))(c_2 g_2(n)) \text{ for } n \geq N_i \\ &\leq (c_1 c_2)(g_1(n) g_2(n)) \text{ for } n \geq N_i \end{aligned}$$

   This satisfies the Big-O definition, where $f_1(n) * f_2(n) = O(g_1(n) * g_2(n))$.

   □

   d. If $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n))$, then $f(n) = \Theta(h(n))$.

   Proof. By definition,
   $f(n) = \Theta(g(n))\ \therefore\ \exists\ a, b\ s.t.\ a * g(n) \leq f(n) \leq b * g(n),\ n \geq n_1$ (eq. 1)
   and
   $g(n) = \Theta(h(n))\ \therefore\ \exists\ c, d\ s.t.\ c * h(n) \leq g(n) \leq d * h(n),\ n \geq n_2$ (eq. 2)
   Since eq. 1 and eq. 2 hold for all $n \geq n_1$ and $n \geq n_2$, respectively, both must hold for $n \geq max(n_1, n_2)$. Therefore,

   $$a * g(n) \leq f(n) \leq b * g(n)$$

$$a * c * h(n) \leq f(n) \leq b * d * h(n)$$

For every $n \geq max(n_1, n_2)$. Hence, $f(n) = \Theta(h(n))$.

□

2. Master Theorem
   a. T(n)=625 T(n/25) + $\Theta(n^{1.75})$
      i. a=625 b= 25 E=1.75
      ii. The Critical factor is C= $log_{25}625$ =2
      iii. Because C > E then that means we are in **Case 1**
      iv. T(n)= $\Theta(n^c) = \Theta(n^2)$

   b. T(n)=$4^{3.5}$ T(n/4) + $\Theta(n^{3.5} log^2 n)$
      i. a=$4^{3.5}$ b= 4 E=3.5
      ii. The Critical factor is C= $log_4 4^{3.5}$ =3.5
      iii. Because C =E then that means we are in **Case 3**
      iv. T(n)= $\Theta(f(n)\ log^{k+1} n)$
      v. Therefore T(n)= $\Theta(n^{3.5} log^3 n)$

   c. T(n)=5 T(n/25) + $\Theta(n)$
      i. a=5 b= 25 E=1
      ii. The Critical factor is C= $log_{25}5$ = .5
      iii. Because C <E then that means we are in **Case 2**
      iv. T(n)= $\Theta(f(n))$
      v. Therefore T(n)= $\Theta(n)$

3. Analyzing Recursive Algorithms
   a. T(n)=3 T(2n/3) + $\Theta(n^{2.709...})$
      i. a=3 b= 3 E=2.709...
      ii. The Critical factor is C= $log_3 3$ =1
      iii. Because C <E then that means we are in **Case 2**
      iv. T(n)= $\Theta(f(n))$
      v. Therefore T(n)= $\Theta(n^{2.709...})$
      vi. While the Stooge sort has the worst case of $\Theta(n^{2.709...})$, selection sort has a worst case of $\Theta(n^2)$ which is still better than stooge sort especially in the long run when you are dealing with a lot more data.

   b. $T(n) = 3T(\frac{n}{3}) + \Theta(n)$
      i. $a = 3,\ b = 3,\ E = 1$

ii. $C = log_3 3 = 1$

iii. $C = E$ $\therefore$ **Case 3**:

1. $T(n) = \Theta(n^{log_b a} log^{k+1} n) = \Theta(nlogn)$

c. $T(n) = 2T(\frac{n}{3}) + \Theta(n)$

i. $a = 2, b = 3,\ E = 1$

ii. $C = log_2 3 = 1.585$

iii. $C > E$ $\therefore$ **Case 1:**

1. $T(n) = \Theta(n^C) = \Theta(n^{log_2 3})$