

CSCI-2725 Homework 1

Yitao Tian

2023-01-21

1. **Problem 1:** Give the Big O estimate of the following functions.

(a) As defined, $f \in O(n^5)$. Given a function $f : \mathbb{N} \rightarrow \mathbb{N}$, where

$$\begin{aligned} f(n) &= 5n^5 - 4n^4 + 3n^3 + \sqrt{2}n^2 + n - 1 \\ (\text{which is in turn}) &\leq 5n^5 + 3n^5 + \sqrt{2}n^5 + 1n^5 \\ (\text{which is in turn}) &= (5 + 3 + \sqrt{2} + 1)n^5 \\ &\therefore f \in O(n^5). \end{aligned}$$

(b) As defined, $f \in O(n^5)$. Given a function f , where

$$\begin{aligned} f(n) &= (n^3 - (\log(n))^3)(n^2 - \log(n)) + 11n^3 \\ &= n^3n^2 - n^3\log(n) - (\log(n))^3n^2 + (\log(n))^4 + 11n^3 \\ &= n^5 - n^3\log(n) - n^2(\log(n))^3 + (\log(n))^4 + 11n^3 \\ &\leq n^5 + (\log(n))^4 + 11n^3 \\ &\leq 1n^5 + 1n^5 + 11n^5 \\ &= (1 + 1 + 11)n^5 \\ &\therefore f \in O(n^5). \end{aligned}$$

(c) As defined, $f \in O(n^2)$. Given a function f , where

$$\begin{aligned}
 f(n) &= \frac{5n^4 + 10n^3 - 100n^2 - n - 1}{6n^2} \\
 &= \frac{5}{6}n^2 + \frac{10}{6}n - \frac{100}{6} - \frac{1}{6n} - \frac{1}{6n^2} \\
 &\leq \frac{5}{6}n^2 + \frac{10}{6}n \\
 &\leq \frac{5}{6}n^2 + \frac{10}{6}n^2 \\
 &= \left(\frac{5}{6} + \frac{10}{6}\right)n^2 \\
 \therefore f &\in O(n^2).
 \end{aligned}$$

(d) As defined, $f \in O(n^2 \log(n))$. Given a function f , where

$$\begin{aligned}
 f(n) &= \log(n^3 + n + 10) + n^2 \log(n + 4) \\
 &\leq \log(1n^3 + 1n^3 + 10n^3) + n^2 \log(1n + 4n) \\
 &= n^2 \log(5n) + \log(12n^3) \\
 &= n^2(\log(5) + \log(n)) + \log(12) + \log(n^3) \\
 &= \log(5)n^2 + n^2 \log(n) + \log(12) + 3 \log(n) \\
 &\leq (1 + \log(5) + \log(12) + 3)n^2 \log(n) \\
 \therefore f &\in O(n^2 \log(n)).
 \end{aligned}$$

(e) As defined, $f \in O(n^2 \log(n)^2)$. Given a function f , where

$$\begin{aligned}
 f(n) &= (n \log(n) + 1)^2 + (\log(n) + 1)(n^2 + 1) \\
 &= n^2(\log(n))^2 + 2n \log(n) + n^2 \log(n) + \log(n) + n^2 + 2 \\
 &\leq (1 + 2 + 1 + 1 + 1 + 1 + 1)n^2 \log(n)^2 \\
 \therefore f &\in O(n^2 \log(n)^2).
 \end{aligned}$$

(f) As defined, $f \in O(\log(n))$. Given a function f , where

$$\begin{aligned}
 f(n) &= \log((5n^5 + 7n^3 + 10)^2(3n^3 + 4n + 10)) \\
 &= \log((25n^{10} + 70n^8 + 100n^5 + 49n^6 + 140n^3 + 100) \\
 &\quad \times (3n^3 + 4n + 10)) \\
 &= \log(75n^{13} + 310n^{11} + 250n^{10} + 427n^9 + 1000n^8 \\
 &\quad + 196n^7 + 1310n^6 + 1000n^5 \\
 &\quad + 560n^4 + 1700n^3 + 400n + 10000) \\
 &\leq \log((75 + 310 + 250 + 427 + 1000 + 196 + 1310 \\
 &\quad + 1000 + 560 + 1700 + 400 + 10000)n^{13}) \\
 &= \log(75 + 310 + 250 + 427 + 1000 + 196 + 1310 \\
 &\quad + 1000 + 560 + 1700 + 400 + 10000) + 13 \log(n) \\
 &\leq (\log(75 + 310 + 250 + 427 + 1000 + 196 + 1310 \\
 &\quad + 1000 + 560 + 1700 + 400 + 10000) + 13) \log(n) \\
 \therefore f &\in O(\log(n)).
 \end{aligned}$$

2. **Problem 2:** Find constants c and n_0 such that the given statement is true.

(a) Show, for a function f , where $f(n) = n^2 + 10$, $f \in O(n^3)$:

$$\begin{aligned}
 \because f(n) &= n^2 + 10 \\
 &\leq 1n^2 + 10n^2, n_0 = 1 \\
 &= 11n^2 \\
 \therefore \text{ if } c = 11 \wedge n_0 = 1, &\text{ then } f \in O(n^3). \quad \blacksquare
 \end{aligned}$$

(b) Show, for a function f , where $f(n) = n!$, $f \in O(n^n)$:

$$\begin{aligned}
 \because f(n) = n! &\triangleq \prod_{i=1}^n i \\
 &= \underbrace{(n)(n-1) \cdots (2)(1)}_{n \text{ factors}} \\
 &\leq \underbrace{(n)(n) \cdots (n)(n)}_{n \text{ factors}}, n_0 = 1 \\
 &= 1^n \\
 \therefore \text{ if } c = 1 \wedge n_0 = 1, &\text{ then } f \in O(n^n). \quad \blacksquare
 \end{aligned}$$

(c) Show, for a function f , where $f(n) = 3n^3 + 7n + 10$, $f \in \Theta(n^3)$:

$$\begin{aligned}
 &\because f(n) = 3n^3 + 7n + 10 \\
 &\leq 3n^3 + 7n^3 + 10n^3, n_0 = 1 \\
 &= 20n^3 \\
 &\therefore \text{ if } c_1 = 20 \wedge n_0 = 1, \text{ then } f \in O(n^3). \\
 &\because f(n) = 3n^3 + 7n + 10 \\
 &\geq 3n^3, n_0 = 1 \\
 &\therefore \text{ if } c_2 = 3 \wedge n_0 = 1, \text{ then } f \in \Omega(n^3). \\
 &\therefore \text{ if } f \in O(n^3) \wedge f \in \Omega(n^3), \text{ then } f \in \Theta(n^3) \\
 &\therefore \text{ if } c_1 = 20 \wedge c_2 = 3 \wedge n_0 = 1, \text{ then } f \in \Theta(n^3). \quad \blacksquare
 \end{aligned}$$

(d) Show, for a function f , where $f(n) = 1^2 + 2^2 + 3^2 + \dots + n^2$, $f \in O(n^3)$:

$$\begin{aligned}
 &\because f(n) = 1^2 + 2^2 + 3^2 + \dots + n^2 \\
 &= \sum_{i=1}^n i^2 \\
 &= \frac{n(n+1)(2n+1)}{6} \\
 &= \frac{1n + 2n^3 + 3n^2}{6} \\
 &\leq \frac{(1+2+3)n^3}{6}, n_0 = 1 \\
 &= n^3 \\
 &\therefore \text{ if } c = 1 \wedge n_0 = 1, \text{ then } f \in O(n^3). \quad \blacksquare
 \end{aligned}$$

3. **Problem 3:** Find the Big O estimate or worst case complexity of the following functions (given in pseudo-code). For the following code snippets, the work for the problem is contained in code comments. The function algorithms ($T(n)$) are measured in units of assignment operations.

(a) The Big O estimate of the below function is $O(N^3)$.

```

1  int fun (int N) {                                // T(N) =
2      for (i = 0; i < N; i++) {                    // N × (
3          if (x < 10) {
4              for (j = 0; j < N * N; j++) { // (N2 ×
5                  a = a + 10;                    // 1) +
6              } // for (level 3)
7          } // if (2)
8          else if (y < 10) {
9              for (p = 0; p <= N * N; p++) { // ((N2 + 1) ×
10                 b = b + 10;                    // 1) +
11             } // for (level 3)
12         } // elif (level 2)
13         else {
14             for (k = 0; k <= 10; k++) { // (11 ×
15                 c = c + 10;                    // 1))
16             } // for (level 3)
17         } // else (level 2)
18     } // for (level 1)
19     // = 2N3 + 12N assignment operations
20 } // fun (level 0)                                // ∴ T ∈ O(N3)

```

(b) The Big O estimate of the below function is $O(N^2)$.

```

1  int fun (int N) {                                // T(N) =
2      for (i = N; i > 0; i--) {                    // N × (
3          for (j = 0; j < i; j++) {                // i × (
4              a = a + 10;                    // 1))
5          } // for (2)
6      } // for (1)
7      // =  $\frac{1}{2}N(N-1)$  assignment operations
8  } // fun (0)                                // ∴ T ∈ O(N2)

```

(c) The Big O estimate of the below function is $O((\log(N))^2)$.

```

1  int fun (int N) {                                // T(N) =
2      for (i = N; i > 0; i = i / 2) {              // (log(N) + 1) × (
3          j = 1;                                    // 1+
4          while (j < N) {                            // log(N) × (
5              a = a + 10;                            // 1+
6              j = 2 * j;                             // 1))
7          } // while (2)
8      } // for (1)
9      // = 2 log(N)2 + 3 log(N) + 1 assignment operations
10 } // fun (0)
11 // ∴ T ∈ O((log(N))2)

```

(d) The Big O estimate of the below function is $O(N(\log(N))^2)$.

```

1  int fun (int N) {                                // T(N) =
2      i = N;                                        // 1+
3      while (i > 0) {                                // (log(N) + 1) × (
4          j = 1;                                    // 1+
5          while (j < N) {                            // log(N) × (
6              k = 0;                                // 1+
7              while (k < N) {                        //  $\frac{1}{2}N \times ($ 
8                  k = k + 2;                         // 1)+
9              } // while (3)
10             j = j * 2;                             // 1)+
11         } // while (2)
12         i = i / 2;                                 // 1)
13     } // while (1)
14     // =  $\frac{1}{2}N \log(N)^2 + 2 \log(N)^2 + 4 \log(N) + \frac{1}{2}N \log(N) + 3$ 
15     // ↪ assignment operations
16 } // fun (0)
17 // ∴ T ∈ O(N(log(N))2)

```