Prof. Pietro S. Oliveto

CS217 Data Structures and Algorithm Analysis
Fall term 2024

Exercise Sheet 2

Handout: September 24th — Deadline: October 1st, 4pm

Question 2.1 (0.5 marks)

Express the following running times in Θ-notation. Justify your answer by referring to the definition of Θ (i. e. work out suitable c_1, c_2, n_0).

a) $3n^2 + 5n - 2$

b) 42

c) $4n^2 \cdot (1 + \log n) - 2n^2$

Question 2.2 (0.5 marks)

(a) Indicate for each pair of functions $f(n), g(n)$ in the following table whether $f(n)$ is O, o, Ω, ω, or Θ of $g(n)$ by writing "yes" or "no" in each box.

$f(n)$	$g(n)$	O	o	Ω	ω	Θ
$\log n$	$\sqrt{n}$					
n	$\sqrt{n}$					
n	$n \log n$					
n^2	$n^2 + (\log n)^3$					
2^n	n^3					
$2^{n/2}$	2^n					
$\log_2 n$	$\log_{10} n$					

Hints: the book states that every polynomial of $\log n$ grows strictly slower than every polynomial n^ε, for constant $\varepsilon > 0$. For example, $(\log n)^{100} = o(n^{0.01})$. Likewise, every polynomial grows slower than every exponential function 2^{n^ε}, for example $n^{100} = o(2^{n^{0.01}})$.

To convert the base of a logarithm, use $\log_x(n) = \log_y(n) / \log_y(x)$.

Question 2.3 (0.5 marks)

State the number of "foo" operations for each of the following algorithms in Θ-notation. Pay attention to indentation and how long loops are run for. Justify your answer by stating constants $c_1, c_2, n_0 > 0$ from the definition of $\Theta(g(n))$ in your answer.

Example: Line 1 is executed once and line 3 is executed $n - 4$ times. So the number of foos is $1 + n - 4 = n - 3 = \Theta(n)$ as $c_1 n \le n - 3 \le c_2 n$ for all $n \ge n_0$ when choosing, say, $n_0 = 6, c_1 = 1/2, c_2 = 1$.

Example Algorithm

```
1: foo
2: for i = 1 to n − 4 do
3:     foo
```

Algorithm A

```
foo
for i = 1 to n do
    for j = 1 to n−2 do
        foo
        foo
        foo
```

Algorithm B

```
foo
for i = 1 to n do
    foo
for i = 1 to n/2 do
    foo
    foo
```

Algorithm C

```
foo
for i = 1 to n do
    for j = 1 to i do
        foo
    foo
foo
```

Question 2.4 (0.5 marks)

Recall from Lecture 2 that a statement like $2n^2 + \Theta(n) = \Theta(n^2)$ is true if *no matter how the anonymous functions are chosen on the left of the equal sign, there is a way to choose the anonymous functions on the right of the equal sign to make the equation valid.* You might want to think of the $\Theta(n)$ on the left-hand side being a placeholder for some (anonymous) function that grows as fast as n.

For each of the following statements, state whether it is true or false. Justify your answers.

1. $O(\sqrt{n}) = O(n)$
2. $n + o(n^2) = \omega(n)$
3. $3n \log n + O(n) = \Theta(n \log n)$

Also, explain why the statement "The running time of Algorithm A is at least $O(n^2)$" is meaningless.

Question 2.5 (0.5 marks)

The following algorithm computes the product C of two $n \times n$ matrices A and B, where $A[i, j]$ corresponds to the element in the i-th row and the j-th column.

Matrix-Multiply(A, B)

```
for i = 1 to n do
    for j = 1 to n do
        C[i, j] := 0
        for k = 1 to n do
            C[i, j] := C[i, j] + A[i, k] · B[k, j]
return C
```

Give the running time of the algorithm (number of operations in a RAM machine) in Θ-notation. Justify your answer. Feel free to use the rules on calculating with Θ-notation from the lecture.

Programming Question 2.6 (0.25 marks)

Implement Matrix-Multiply(A,B).