# COMP 3761 - Algorithm-analysis-and-design

## Chapter 2.1 Page 50 Question 2

1. **Consider the definition-based algorithm for adding two n × n matrices. What is its basic operation? How many times is it performed as a function of the matrix order n? As a function of the total number of elements in the input matrices?**

C = A + B

where c = a ​+ b

n x n = n^2

C = n^2 + n^2 = 2n^2 -> it will still be n^2

The basic operation is the addition of two numbers and the operation performed is n^2 times even though the total number of elements in two input matrices is 2n^2

The number of basic operations is n^2 regardless of the total number of elements in the input matrices (which is 2n^2 for the two n × n matrices), because the operation corresponds directly to the number of elements in the resulting matrix C.

1. **Answer the same questions for the definition-based algorithm for matrix multiplication.**

n^2 x n = n^3

n^3 - n^2

the number of basic operations is n^3 for n^3 - n^2

Its basic operation is multiplication of two numbers and the operation performed is n^3 because the overall time complexity is O(n^3) and the overall time complexity is O(n^3) which corresponds to the number of basic operations in matrix multiplication.

The number of basic operations is n^3, and it is independent of the total number of elements in the input matrices (which is 2n^2 in total), since the total number of operations depends primarily on the size of the output matrix and the need to compute each element via a summation over 𝑛

## Chapter 2.2 Page 60 Question 5

List the following functions according to their order of growth from the lowest to the highest:

ln^2(n) < 5lg(n+100)^10 < n^(3/2) < 0.001n^4 + 3n^3 + 1 < 3^n < 2^(2n) < (n-2)!

## Chapter 2.3 Page 68 Question 5, 6

### Question 5

line 1  ALGORITHM Secret(A[0..n − 1])

            //Input: An array A[0..n − 1] of n real numbers

line 2      minval ← A[0]; maxval ← A[0]

line 3      for i ← 1 to n − 1 do

line 4          if A[i] < minval

line 5              minval ← A[i]

line 6          if A[i] > maxval

line 7              maxval ← A[i]

line 8      return maxval − minval

Answer questions (a)–(e) of Problem 4 about this algorithm.

1. **What does this algorithm compute?**

the difference between the largest and smallest number

1. **what is its basic operation?**

line 4 if A[i] < minval

line 6 if A[i] > maxval

both are the operation and depending which one runs more often

1. **how many times is the basic operation executed**

There are two comparisons

A[i] < minval

A[i] > maxval

The total number of basic operations is 2 x (n - 1) = 2n - 2 = n

1. **what is the efficiency class of this algorithm?**

The number of basic operations is 2n-2 which is linear in terms of n.

the algorithm performs a constant amount of work (2 comparisons) for each element in the array (except for the first one)

The time complexity is O(n) where n is the size of the input array

The efficiency class of the algorithm is O(n)

1. **suggest an improvement, or a better algorithm altogether, indicate its efficiency class. If you cannot do it, try to prove that, in fact, it cannot be done.**

The algorithm is already really efficient for the task it performs. To compute the range of an array, you need to examine each element at least once to determine the minimum and maximum values. Hence, this algorithm performs a linear scan of the array which is optimal in terms of time complexity for this specific task.

### Question 6

line 1  ALGORITHM Enigma(A[0..n − 1, 0..n − 1])

            //Input: A matrix A[0..n − 1, 0..n − 1] of real numbers

line 2      for i ← 0 to n − 2 do

line 3          for j ← i + 1 to n − 1 do

line 4              if A[i, j ] =/= A[j, i]

line 5                  return false

line 6      return true

1. **What does this algorithm compute?**

This algorithm computes to see if the matrix is symmetric or not. If it is not symetric then the algorithm returns false. If the matrix is symmetric then it returns true.

1. **What is its basic operation?**

The basic operation is line 4 if A[i, j ] =/= A[j, i]

1. **How many times is the basic operation executed?**

(n-1)-(i+1)+1 = n-i-1

(n-2)-0+1 = n-1

((n-2+1)-0+1)(n-2)/2 = n(n-1)/2

(n-2)-0+1 = n-1

The total number of basic operations is n(n-1)/2 and the time complexity is O(n^2)

1. **What is the efficiency class of this algorithm?**

The efficiency of the algorithm is O(n^2)

1. **Suggest an improvement, or a better algorithm altogether, and indicate its efficiency class. If you cannot do it, try to prove that, in fact, it cannot be done.**

The algorithm performs a comparison for each pair of elements A[i,j] and 𝐴[𝑗,𝑖], and checks for symmetry. The algorithm is already optimal in terms of its complexity of O(n^2)

no improvement is needed