COM1029 Data Structures and Algorithms

Jim Lam

February 5, 2024

Contents

1 Algorithms **1**
1.1 History of Algorithms 1
1.2 Algorigthm analysis 1
1.2.1 Time complexity 1
1.2.2 Dominant term 2
1.2.3 Big O notation 2
1.2.4 Upper bound (Pessimistic view) 2

1 Algorithms

1.1 History of Algorithms

My boy Muhammad ibn Musa al-Khwarizmi was the first to write a book on the systematic solution of linear and quadratic equations. He was a Persian mathematician, astronomer, and geographer during the Abbasid Caliphate, a scholar in the House of Wisdom in Baghdad. He was the first to introduce the concept of algorithm to the Western world. The word algorithm comes from the Latin word *algorismus*, which is a Latinization of his name. He is also known for his work on algebra, which is derived from the Arabic word *al-jabr*.

1.2 Algorigthm analysis

1.2.1 Time complexity

Example (find the average of an array of n integers)

```
% find the average of an array of n integers
    int sum = 0;                // 1
    for (int i = 0; i < n; i++) { // n
        sum += i;               // n
    }
    return sum / n;             // 1
```

Quadratic example:

```
% find the sum of an array of n integers
    int[][] a = someArray;       // 1
    int sum = 0;                     // 1
    for (int i = 0; i < len(a); i++) {   // n
        for (int j = 0; j < len(a[i]); j++) { // n^2
            sum += a[1][j];              // n^2
        }
    }
    return sum;                      // 1
```

1.2.2 Dominant term

The dominant term is the one with highest power (degree) in a function

Example: the cubic function f(N) = 10N3 + N2 + 40N + 80

- 10N3 is the dominant term (among 10N3 , N2 , 40N and 80)
- When we look at f(1000) = 10,001,00,080, we see that 10,000,000,000 is due to the 10N3 term
- If we use the approximation f(N) = 10N3, then we would only be 0.01% out

The value of f(N) is largely determined by the dominant term, for sufficiently large N
- The meaning of ‘sufficiently large’ varies according to the function

1.2.3 Big O notation

Compared to evaluating the dominant term, Big O notation is a more general way of expressing the time complexity of an algorithm. It is a way of expressing the upper bound of a function.

Example: $f(N) = 10N^3 + N^2 + 40N + 80$
- f(N) is O(N3)
- f(N) is O(N2)
- f(N) is O(N)
- f(N) is O(1)

1.2.4 Upper bound (Pessimistic view)