# PH502: Scientific Programming Concepts

Irish Centre for High End Computing (ICHEC)

September 23, 2020

## Overview

- In this lecture we will consider the properties of algorithms.
- Also how to evaluate them.
- We will consider one aspect in detail, long long they take.

## Properties of Algorithms

- Finiteness: Terminates after a finite number of steps. Otherwise it cannot be called as computational method.
- Definiteness: Each step is unambiguous. (Each step must be precisely defined.)
- Effectiveness: Consist of basic instructions that are realisable/computable. Its operations must all be sufficiently basic that they can in principle be done exactly and in a finite length of time by someone using pencil and paper.
- Set of Inputs: Quantities that are given to the algorithm initially before it begins, or dynamically as it runs, i.e., a, b,
- Set of Outputs: Quantities that have a specified relation to the inputs, i.e., gcd(a, b).

## Properties of Algorithms

- Adaptability: Can it be modified or evolved over time to meet changes?
- Reusability: Can it be used as a component in different systems or in different application domains?
- Correctness: Does it solve the problem for all possible inputs?
- Efficiency:
  - Space Complexity: How much memory taken? - minimal use of computational resources
  - Time Complexity: How long it takes to solve? - fast runtime

There are many algorithms, each with different features, to solve the same problem. As we shall see later many of the better ones are freely available.

## Why study Algorithms?

"Having a solid base of algorithmic knowledge and technique is one characteristic that separates the truly skilled programmers from the novices. With modern computing technology, you can accomplish some tasks without knowing much about algorithms, but with a good background in algorithms, you can do much, much more."

*Introduction to Algorithms (3rd Ed.) MIT Press*

## Algorithm Analysis

- Theoretical study of computer program performance and resource usage.
  - Memory, disk, communication bandwidth, energy spent, but mostly computational time
- Computational time:
  - Depends on input data (sorted or not sorted)
  - Depends on input size (10 versus $10 \times 10^6$)
  - Maximum time given any input of size $n$ (Worst case).
- Which algorithm is the best one to use? - *Depends on the computer*
  - Relative speed (on the same computer)
  - Absolute speed (on different computers)

## Asymptotic Analysis

- No matter what the computer is, we compare the growth rate!
- Which algorithm is more efficient?

| **Algorithm** | **Input 1** | **Input 2** | **Input 3** |
|---|---|---|---|
| **Algorithm 1** | 30s | 35s | 40s |
| **Algorithm 2** | 1s | 4s | 30s |

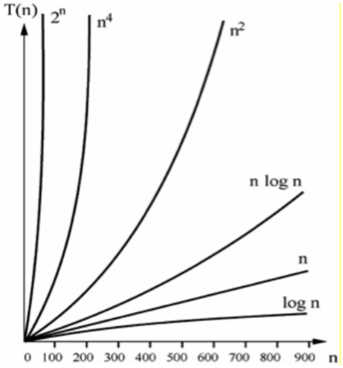
- The efficiency of an algorithm is not tied to the consumption of a resource for a specific input, but to how it responds to changes in input size: Growth Rate.
- How well does the algorithm perform as the input size grows;

$$n \to \infty$$

## Time Complexity

When the input size increases, the runtime may...

- Remain constant: $\Theta(1)$
- Increase logarithmically: $\Theta(logn)$
- Increase linearly: $\Theta(n)$
- Increase quadratically: $\Theta(n^2)$
- Increase cubically etc: $\Theta(n^3)$ or $\Theta(n^4)$ or ... (*Polynomial complexities.*)
- Increase exponentially: $\Theta(2^n)$



from lecture notes by Georgy Gimel'farb