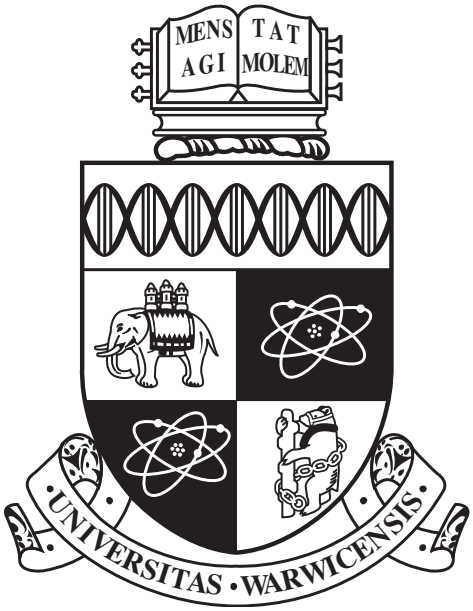
University of Warwick
Department of Computer Science

CS126

Design of Information Structures



Cem Yilmaz
January 14, 2022

1 Analysis of Algorithms

1.1 Classification of a good algorithm

A good algorithm is one that optimises the following:

- Time
- Memory
- Network bandwidth
- Energy consumption

However, the main focus in this module will be the running time, in particular, this $O(x)$

1.2 Running Time

Running time of an algorithm typical grows with the input size. However, for different inputs of the same size the running time of an algorithm can vary. Then, for an input of fixed size n, we have different running times. We can categorise them as the following:

- Average case - the typical running time an algorithm requires and is often very difficult to determine
- Best case - what is the minimum running time of the algorithm and is generally not useful
- Worst case - upper bound on the running time, for any possible input and is more standard to analyse. Our focus is generally this.

1.3 Finding the running time

1.3.1 Experimental Analysis

The first method to use is experimental analysis. For this, we use computer and run simulations. For this, we write a program implementing the algorithm and run the program with inputs of varying size and composition, noting the time needed. However, there are limitations:

- It is necessary to implement the algorithm. Sometimes this can be impossible.
- Need to make sure that we have considered all kinds of inputs. This sometimes cannot be possible and otherwise it would not be indicative of running time
- Different algorithms may run different in different systems due to different features. It can be especially worse for different hardware.

1.3.2 Theoretical Analysis

The second method is theoretical analysis. For this, we use pen and paper. We use a high-level description of the algorithm instead of an implementation. We then characterise running time as a function of the input size n which takes in account all possible inputs. This would indeed allow us to evaluate the speed of an algorithm independent of the hardware or software environment. A good way to do high level description is to use pseudo-code. We also assume that the algorithm runs in an idealised machine. We assume simple memory hierarchy that is unbounded, infinite precision in arithmetic operations etc.