Algorithmics - Asymptotic Notaion

Harry leslie

February 2023

1 Introduction

Algorithmics is the study to get a further understanding how algorithms and data structures work This document will be on the first lecture in the COMP1201 at the University of Southampton.

2 Lecture 1 - Asymptotic Notation

To summarise this lecture main take away is to understand what the bounds are for different functions

You had to understand firstly what Big O, θ, Ω are, but also how to solve proofs with them. Remember not to get confused with best/worst/average case analysis and bound analysis they are not the same thing

You also have to do some algorithm analysis of best, worst and average cases. This takes some practice so I will go over some algorithms to help with understanding. For algorithm analysis check the asymptotic notation folder

2.1 Big O

Big O refers to the upper bound of a function and is formally known as:

$$if\ g(n)\ \ is\ \ O(f(n))\ \Rightarrow\ g(n)\ \leq\ c*f(n),\ s.t.\ c>0,\ \ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

This looks very confusing but if you try some examples it would actually make sense.

2.2 Example 1

$$n+1\ is\ O(n^2)$$

Using the definition of big O, you can say that:

$$\Rightarrow\ n+1\ \leq\ c*n^2,\ s.t.\ c>0,\ \ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

Remember that this is asymptotic notation, meaning that as n approaches infinity which will have a higher value

if this is confusing try thinking about whether you can do

$$\lim_{n\to\infty} \frac{g(n)}{c(f(n))} = 0$$

With this example you can let $c = 2$ and in this case:

$$\Rightarrow\ n+1 \leq 2*n^2,\ s.t.\ c>0, \forall n > N,\ N\ \epsilon\ \mathbf{N}$$

which is true for all values of $n \geq 1$

Therefore you have proven that $n+1\ is\ O(n^2)$

$$\lim_{n\to\infty} \frac{n+1}{2n^2} = 0$$

2.3 Example 2

$$n+1\ is\ O(log(n^n))$$

Using the definition of big O, you can say that:

$$\Rightarrow\ n+1 \leq c*log(n^n),\ s.t.\ c>0,\ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

$$\Rightarrow\ n+1 \leq c(nlog(n)),\ s.t.\ c>0,\ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

With this example you can let $c = 2$ and in this case:

$$\Rightarrow\ n+1 \leq 2*(nlog(n)),\ s.t.\ c>0,\ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

which is true for all values of $n \geq 1$

Therefore you have proven that $n+1\ is\ O(log(n^n))$

2.4 Big Ω

Using the definition of Big Omega:

$$if\ g(n)\ \ is\ \ \Omega(f(n))\ \Rightarrow\ g(n) \geq c*f(n),\ s.t.\ c>0,\ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

2.5 Example 3

$$n+1\ is\ \Omega(log(n)\ +\ n)$$

$$\Rightarrow\ n+1 \geq c(n+log(n)),\ s.t.\ c>0,\ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

With this example you can let $c = \frac{1}{3}$ and in this case:

$$\Rightarrow\ n+1 \geq \frac{1}{3}*(n+logn(n)),\ s.t.\ c>0,\ \ n>N,\ N\ \epsilon\ \mathbf{N}$$

which is true for all values of $n \geq 1$

Therefore you have proven that $n + 1 \; is \; \Omega(n + log(n))$

This is true because you know that n + n $\geq$ n + log(n) so therefore if you half the value on the right then if you divide the values by a third then it must be less the the left hand side

2.6 Example 4

Prove n^{10} is not $\Omega(2^n)$

Proof by Contradiction:

$$\Rightarrow \; n^{10} \; \geq \; c(2^n), \; s.t. \; c > 0, \quad n > N, \; N \; \epsilon \; \mathbf{N}$$

However this is not possible as:

$$\lim_{n \to \infty} \; \frac{n^{10}}{2^n} \; = \; 0$$

but

$$\frac{n^{10}}{2^n} \geq c > 0$$

so therefore you have proven that n^{10} is not $\Omega(2^n)$

2.7 Big θ

Using the definition of Big Theta:

$$if \; g(n) \;\; is \;\; \theta(f(n)) \; \Rightarrow c * f(n) \leq g(n) \leq \; d * f(n), \; s.t. \; c, d > 0, \quad n > N, \; N \; \epsilon \; \mathbf{N}$$

2.8 Exercise 1

Prove f(n) is θ(g(n)) is equivalent to f(n) is O(g(n)) and g(n) is O(f(n)).