Informatics 2 – Introduction to Algorithms and Data Structures

Tutorial 1: Asymptotic Notation

This tutorial is designed to help you become fluent in asymptotic notation, which was introduced in Lectures 3 and 4, and which will be used throughout the rest of the course. The emphasis this week is on the basic mathematical machinery, touching only lightly on algorithms in the last question. We'll see a lot more applications to algorithms in next week's tutorial and later in the course.

Please work through as much of the sheet as you can in advance of the tutorial, and come prepared to ask questions on any points where you encounter difficulties.

Questions marked $\star$ may be more challenging than the others.

1. First, some practice in working intuitively with growth rates. Here you should try to give informal justifications for your answers, though you are not required to present them with full mathematical rigour.

   Recall from the lectures the following set of functions representing some commonly arising growth rates. Here $\lg n$ means the logarithm of n to base 2.

$$\begin{array}{lllllllll} f_0(n) & = & 1 & f_1(n) & = & \lg n & f_2(n) & = & \sqrt{n} \\ f_3(n) & = & n & f_4(n) & = & n \lg n & f_5(n) & = & n^2 \\ f_6(n) & = & n^3 & f_7(n) & = & 2^n & f_8(n) & = & 2^{2^n} \end{array}$$

   For each of the following five functions g, identify a function f_i from the above list such that $g = \Theta(f_i)$. Justify your answers as clearly as you can.

   (a) $g(n) = n(n+1)(2n+1)/6$.

   (b) $g(n) = n$ div 57 (integer division, rounding down)

   (c) $g(n) = n$ mod $57 + 1$

   (d) $g(n) = n \lg n + (\lg n)^3 + e^{-n}$. You may assume here that $\lg n = o(\sqrt{n})$.

   (e) $\star$ Where would the *factorial* function fit into this picture? Does $n!$ have the same growth rate as one of the above functions f_i? Or does it fall between f_i and f_{i+1} for some i?

2. The next stage is to learn to argue with full rigour from the definitions of $o(\cdot)$, $O(\cdot)$, etc. This may take a while to master, but see how you get on with the following examples at this stage.

Recall from the lecture slides that '$f \in o(g)$' (also written as '$f = o(g)$') means that

for all $c > 0$, there exists N such that for all $n \geq N$ we have $f(n) < cg(n)$,

and '$f \in O(g)$' (or '$f = O(g)$') means that

there exist $C > 0$ and N such that for all $n \geq N$ we have $f(n) \leq Cg(n)$.

(a) Show directly from the definition that $100n^3 = o(n^4)$.

(b) Show that if r, s are any *real* numbers with $0 \leq r < s$, then $n^r = o(n^s)$.

(c) Writing 'lg' for log to base 2 and 'ln' for log to base e, show that $\ln n = O(\lg n)$. Deduce that $\lg n = \Theta(\ln n)$. (This is an important fact: it says that it makes no mathematical difference whether we write e.g. $O(n \lg n)$ or $O(n \ln n)$.)

(d) Is it likewise true that $2^n = \Theta(e^n)$? Justify your answer.

Note: In this course, we'll allow you to assume the following mathematical facts without proof:

- Polynomial functions grow more slowly than exponential ones: for any k and any $r > 1$, we have $n^k = o(r^n)$.
- Logs grow more slowly than square roots, cube roots etc.: for any $k \geq 1$ we have $\lg n = o(n^{1/k})$.

3. Recall the methods you learned at school for addition, long multiplication and long division. For each of these, *informally* analyse the asymptotic worst-case runtime when both inputs have at most n decimal digits. (E.g. is it $\Theta(n)$, or $\Theta(n \lg n)$, or something else?) You may take 'time' to mean the number of times you have to write a symbol on the page.

   [*Note:* This is a more 'fine-grained' level of analysis than we'll usually be concerned with in this course. For many purposes, we'll consider additions and multiplications as 'atomic' operations taking just a single step.]