Algorithms. Assignment 1

August Wiklund

September 2024

2 Asymptotic order

Take the following list of functions and arrange them in ascending order of growth rate. That is, if function $f(n)$ comes before function $g(n)$ in your list, then it should be the case that $f(n)$ is $\mathcal{O}(g(n))$.

$$
\begin{aligned}
f_1(n) &= 4^n \\
f_2(n) &= n^{1.5} \\
f_3(n) &= 2^{2^n} \\
f_4(n) &= n^{100} \\
f_5(n) &= 2^{n^2} \\
f_6(n) &= n(\log n)^2 \\
f_7(n) &= n^{\log n}
\end{aligned}
$$

We can start by taking the logarithms of f_1 - f_7 and get:

$$
\begin{aligned}
\log(f_1(n)) &= n \log 4 \\
\log(f_2(n)) &= 1.5 \log n \\
\log(f_3(n)) &= 2^n \log 2 \\
\log(f_4(n)) &= 100 \log n \\
\log(f_5(n)) &= n^2 \log 2 \\
\log(f_6(n)) &= \log n + 2 \log(\log n) \\
\log(f_7(n)) &= (\log n)^2
\end{aligned}
$$

Now we can easily see that f_6 is the slowest, followed by f_2, f_4, and f_7. Note however that

$$f_7(n) > f_4(n) \implies log(n) > 100 \implies n > x^{100}$$

where x is the base of our logarithm. If we use base 10 we would need to have **a googol** of operations before f_7 outperforms f_4.

After f_7 comes f_1 which leaves us with two final functions,

$$f_3(n) = 2^{2^n}$$

and

$$f_5(n) = 2^{n^2}.$$

As they are both on the form 2^α we can compare the α parts

$$\alpha_3 = 2^n$$

and

$$\alpha_5 = n^2.$$

Here we can once again use the logarithm trick and easily see that

$$\log \alpha_3 = n \log 2 > 2 \log n = \log \alpha_5$$

meaning that the final list becomes:

$$\begin{aligned}
f_6(n) &= n(\log n)^2 \\
f_2(n) &= n^{1.5} \\
f_4(n) &= n^{100} \\
f_7(n) &= n^{\log n} \\
f_1(n) &= 4^n \\
f_5(n) &= 2^{n^2} \\
f_3(n) &= 2^{2^n}.
\end{aligned}$$