Computational Complexity

ian.mcloughlin@gmit.ie

Computational complexity

Length of the input (n).

Number of lookups of the Turing machine state table.

Function: number of lookups versus length of input.

Worst case only.

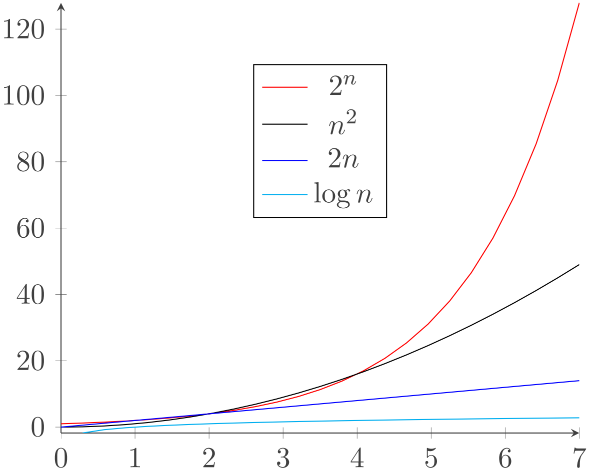
$$f(n) = n \log n$$

Average vs. worst case

Input	Algorithm A	Algorithm B
(1,2,3)	1ms	1ms
(1,3,2)	1ms	5ms
(2,1,3)	2ms	4ms
(2,3,1)	2ms	5ms
(3,1,2)	2ms	5ms
(3,2,1)	10ms	4ms
Average	3ms	4ms
Worst	10ms	5ms

Would you choose Algorithm A or Algorithm B?

Terminology of complexity (graph)



Linear

$$f(n) = a_0 + a_1 n$$

How many pairs of shoes does a centipede need?

- Let's say a centipede has 100 feet.
- Then every centipede needs 100 shoes.
- That's 50 pairs of shoes.
- So 2 centipedes need 100 pairs, 3 need 150 pairs, etc.
- So n centipedes need $50n$ pairs of shoes.
- Linearity is familiar, and most people's default assumption.
- You take the input, multiply by a constant, and add another constant.

Polynomial

$$f(n) = a_0 + a_1 n + a_2 n^2 + a_3 n^3 + \dots$$

What is the volume of a cube of side n?

- Suppose we have a cube with sides of length 1 metre.
- The volume of the cube is $1 \times 1 \times 1 = 1$ metres cubed.
- Suppose the cube has sides of length 2 metres instead.
- The volume of the cube is $2 \times 2 \times 2 = 8$ metres cubed.
- In general, for sides of length n, the volume is n^3.

Exponential

$$f(n) = a^n$$

How many numbers can we represent with n bits?

- Consider the case of four bits – imagine four placeholders

?	?	?	?

- Each placeholder can contain either 0 or 1.
- There are $2 \times 2 \times 2 \times 2 = 2^4 = 16$ different numbers.
- Add another bit, how many numbers is it now?
- It's $2 \times 2 \times 2 \times 2 \times 2 = 2^5 = 32$.
- Generally n bits can represent 2^n numbers.

Logarithmic

$$f(n) = \log_a n$$

How many bits do we need to represent n numbers?

- If we have n bits we can represent 2^n numbers.
- If we want to represent n numbers, how many bits to we need (at a minimum)?
- The inverse operation to exponentiation is logarithm.
- Remember, $a^n = b$ means $\log_a b = n$.

Big-O (Sipser)

Definition

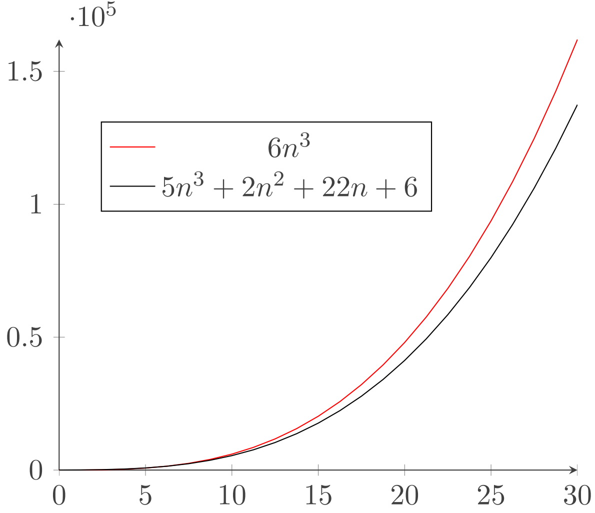
Let f and g be functions $f, g : \mathbb{N} \to \mathbb{R}^+$. We say that $f(n) = O(g(n))$, or f is *big-O* of g, if positive integers c and n_0 exist such that, for every integer n greater than or equal to n_0, $f(n) \leq cg(n)$.

Example

Let f be the function $f(n) = 5n^3 + 2n^2 + 22n + 6$. We'll prove that f is big-O of n^3 ($f = O(n^3)$). Let c be 6 and n_0 be 10. Is the following true, for all n greater than or equal to 10, $5n^3 + 2n^2 + 22n + 6 \leq 6n^3$? Note that as n increases ($n = 10, n = 11, n = 12, \ldots$), $f(n)$ also increases. Also note that $f(10) = 5426$ and $6g(10) = 6000$.

math.mit.edu/ sipser/book.html

Big-O example graph



Smaller values of n