

Algorithm

- Series of steps
- transforms inputs
- has outputs

Efficiency

- Running time
- Memory
- Quality
- Simplicity

Run time

- depends on input
- also depends on data (sorted vs. unsorted) (structured vs unstructured)
- Best/Worst/Average cases

Theoretical Analysis:

- We need to develop a general methodology
- Running time as a function of input size
- independent of environment

Primitive Operations:

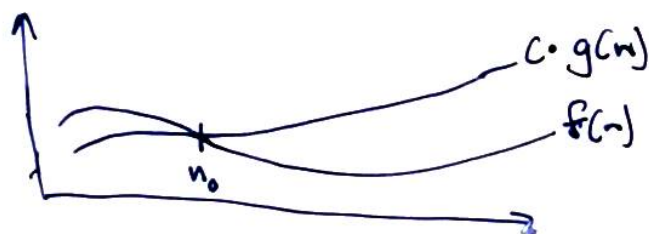
Low-level comparisons from the programming language can be identical in pseudocode

By inspecting pseudocode we can count the number of primitive operations executed by an algorithm

Big-Oh (upper-bound)

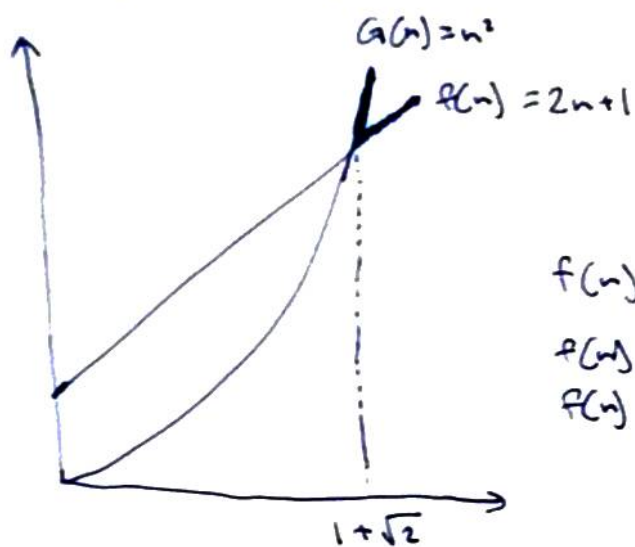
given  $f(n)$  and  $g(n)$ ,  $f(n)$  is  $O(g(n))$  iff there are positive constants  $c$  and  $n_0$  such that

$$\{f(n) \leq c \cdot (g(n)) \text{ for } n \geq n_0 \mid c > 0\}$$



after some point  $n_0$ ,  $c \cdot g(n)$  is always greater than  $f(n)$ .

Graphical Example of



$f(n)$  is  $O(n^2)$

$f(n) \leq c \cdot g(n)$  for  $n \geq n_0$

$f(n) \leq g(n)$  for  $n \geq 1 + \sqrt{2}$

Prove  $60n^2 + 5n + 1$  is  $O(n^2)$

we must find  $c$  and  $n_0$  s.t.

$$60n^2 + 5n + 1 \leq c \cdot n^2 \quad \forall n \geq n_0$$

$$\left. \begin{array}{l} 5n \leq 5n^2 \quad \forall n \geq 1 \\ 1 \leq n^2 \quad \forall n \geq 1 \end{array} \right\} f(n) \leq 60n^2 + 5n^2 + n^2 \quad \forall n \geq 1$$

$$f(n) \leq 66n^2 \quad \forall n \geq 1 \quad \therefore n_0 = 1 \quad c = 66 \quad \therefore O(n^2)$$