Asymptotic Analysis

- way to compare speed of two possible solutions to the same problem.

- Basic idea is to think of the rate of growth of the running time of an algorithm.

  - How much slower does the algorithm gets if we double the input size?

  - we try to ~~focus~~ focus on big inputs.

- for asymptotic analysis, we will first define what is the rate of growth of a function. (Big Oh and related defs)

- In other words, asymptotic analysis is a way to describe the efficiency of algorithms by examining how their performance grows as the input becomes very large. (Trying to get a big picture, ignoring small details)

Key concept of asymptotic analysis will be the definition of growth rates, of functions f and g, and how these growth rates compare.

Big O

Def: Let $f, g : \mathbb{R}^{\geq 0} \longrightarrow \mathbb{R}^{\geq 0}$, we say that f grows no faster than g if there exist constant $c > 0$ and $n_0 \geq 0$ s.t

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

In this case, we write

$$f(n) \text{ is } O(g(n)) \quad \text{or} \quad f(n) = O(g(n))$$

The intuition behind $f(n) = O(g(n))$ is that, for all large enough n, we have $f(n) \leq c \cdot g(n)$

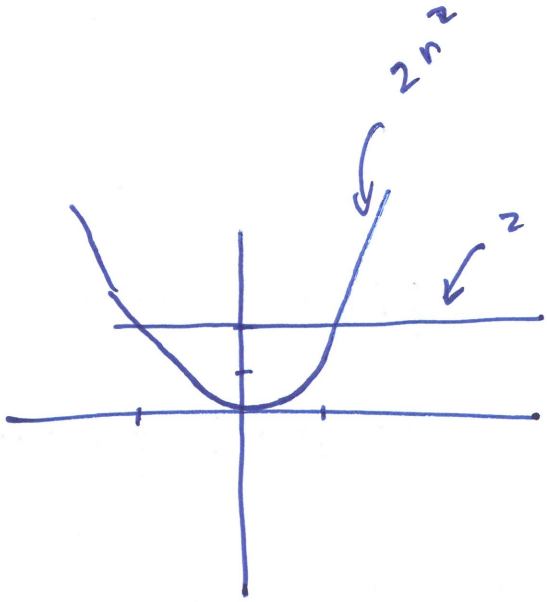
Example: $f(n) = 3n^2 + 2$ is $O(n^2)$

$$f(n) \leq c \cdot n^2$$

Def Big O says

If $f(n) = O(g(n))$, Then

$$\left[\exists c>0, n_0 \geq 0 : \forall n \geq n_0 : f(n) \leq c \cdot g(n)\right]$$



WTS $3n^2 + 2 \leq c \cdot n^2$

$\forall n \geq 1 : 2 \leq 2n^2$

$\forall n \geq 1$

$3n^2 + 2 \leq 3n^2 + 2n^2$

$\forall n \geq 1 : \underbrace{3n^2+2}_{f(n)} \leq 5\underbrace{n^2}_{g(n)}$

pick $c = 5, n_0 = 1$, then

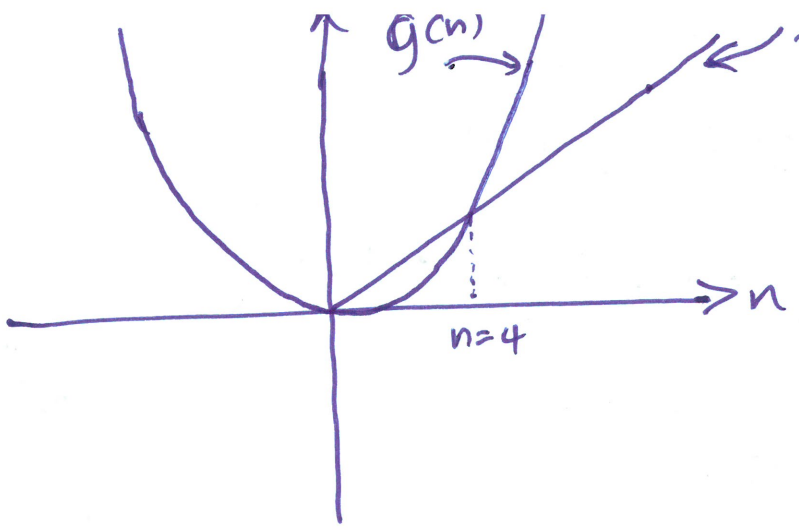
$\forall n \geq 1 : f(n) \leq c \cdot g(n)$

Ex: $f(n) = 4n$

Claim: $f(n) = O(n^2)$ ($f(n) = O(n)$)

we must show a $c>0, n_0 \geq 0$ s.t

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



$f(n) = 4n$

$g(n) = n^2$

pick $c = 1$, $n_0 = 4$

$\forall n \geq 4 : 4n \leq 1 \cdot n^2$

Q: is $4n = O(4n^2)$ yes

is $3n^2 + 2 = O(\frac{1}{2} n^2)$

pick $c = 100$

$\forall n \geq 50$ $3n^2 + 2 \leq 100 \cdot \frac{1}{2} \cdot n^2$

Ex: ~~Is~~ Is $n^3 = O(n^2)$ (no)

can we find $c > 0, n_0 \geq 0$ s.t $\forall n \geq n_0 : n^3 \leq c \cdot n^2$ } we cannot

n^3 is not $O(n^2)$ ← How to prove this?

If $n^3 = O(n^2)$, then

$\exists c > 0, n_0 \geq 0 : \forall n \geq n_0 : f(n) \leq c \cdot g(n)$

since this ~~is~~ is incorrect, the negation of this ~~claim~~ should be true.

$\neg [\exists c > 0, n_0 \geq 0 : \forall n \geq n_0 : f(n) \leq c \cdot g(n)]$

$\neg [\exists x \in S : P(x)] \equiv [\forall x \in s : \neg P(x)]$ *

$\neg [\forall x \in S : P(x)] \equiv [\exists x \in s : \neg P(x)]$ * } Predicate logic theorems.

$[\forall c > 0, n_0 \geq 0 : \neg [\forall n \geq n_0 : f(n) \leq c \cdot g(n)]$

$[\forall c > 0, n_0 \geq 0 : [\exists n \geq n_0 : \neg (f(n) \leq c \cdot g(n))]$

$[\forall c > 0, n_0 \geq 0 : [\exists n \geq n_0 : f(n) > c \cdot g(n)]$

$\forall c > 0, n_0 \geq 0$: I need to show the existence of n s.t

$$f(n) > c \cdot g(n)$$

$$n^3 > c \cdot n^2$$

What if we pick $n = c+1$

$$(c+1)^3 > c \cdot (c+1)^2$$

$$n = \max(n_0, c+1)$$

When $n = \max(n_0, c+1)$, $\forall c > 0, n_0 \geq 0$, $f(n) > c \cdot g(n)$

$\therefore$ $f(n)$ is not $O(n^2)$

Note that

$f(n) = 4n^3 + 4n + 5$

$f(n) = O(n^3)$

$f(n) = O(n^4)$

$f(n) = O(n^5)$

However, the tightest $O(\cdot)$ function for $f(n)$ is $O(n^3)$

Note: When we write the Big O function for $f(n)$, we want to use the tightest bound, as it is more useful.

common distinct functions.

Name	Example f(n)	generic f(n)	tightes $O(\cdot)$
constant	$f(n) = 10$	$f(n) = c,\ c \in \mathbb{R}_{\geq 0}$	$O(1)$
log	$f(n) = 2\log_{10}(n)$		$O(\log n)$
linear	$f(n) = 10 \cdot n$		$O(n)$
n log n	$f(n) = 8n\log_{12}(n)$		$O(n \log n)$
quadratic	$f(n) = 12n^2$		$O(n^2)$
cubic	$f(n) = 20n^3$		$O(n^3)$
deg-k polynomial		$f(n) = a_1 n^k + a_2 n^{k-1} + \dots + a_n \cdot 1$	$O(n^k)$
exponential	$f(n) = 3 \cdot 2^n$	2^n	$O(2^n)$
	$f(n) = 5 \cdot 3^n$	3^n	$O(3^n)$
		$\vdots$	

functions later on this list $\neq O($functions earlier on this list$)$

$$2^n \neq O(n^{100})$$

$$3^n \neq O(n)$$

- However,

functions earlier $= O($functions later on the list$)$

$$n = O(2^n)$$

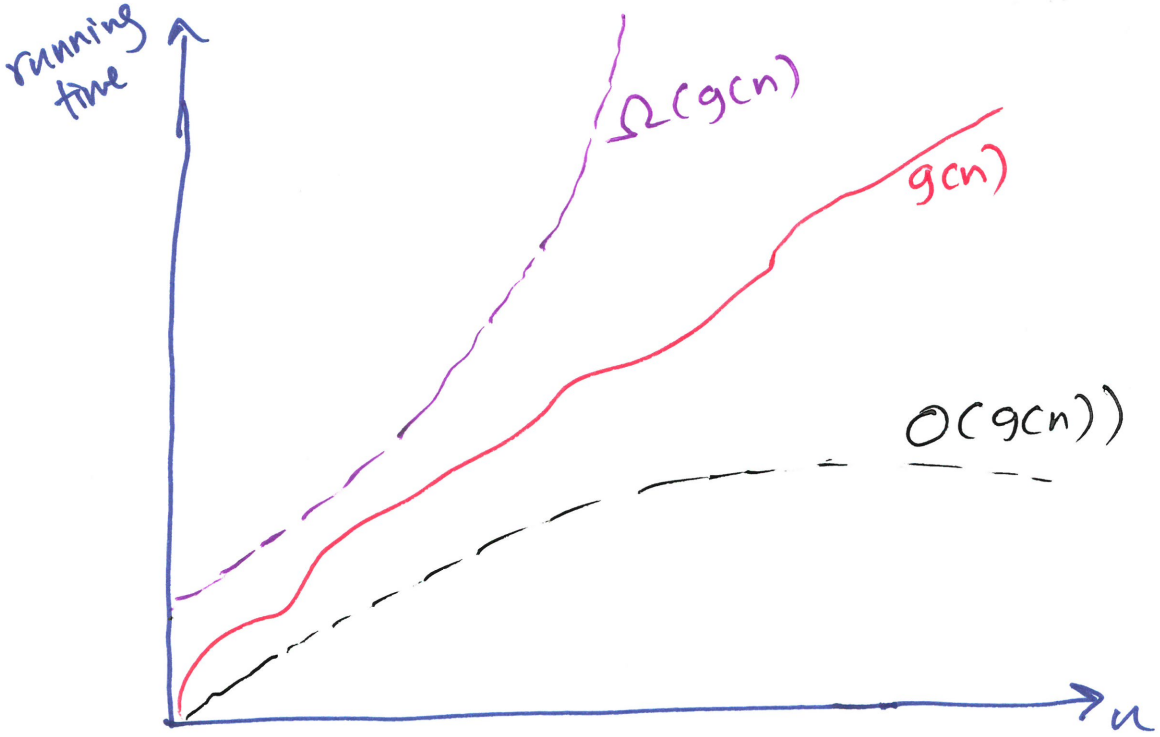
$$n^{100} = O(2^n)$$

$$n^{100} = O(3^n)$$

Other asymptotic notations

Big-Omega (Ω) – f grows no slower than g.

f is $\Omega(g)$ if $\exists d > 0$, $n_0 \geq 0$ s.t $\forall n \geq n_0$: $f(n) \geq d \cdot g(n)$



Big-Theta (Θ)

f is $\Theta(g)$ if $f = O(g)$ and $f = \Omega(g)$

In computer science this is what we are interested in.

Properties of Big O

Lemma 6.2 Asymptotic equivalence of max and sum

we have,

$$f(n) = O(g(n) + h(n)) \iff f(n) = O(\max(g(n), h(n)))$$