2 Intro. To the Analysis of Algorithms

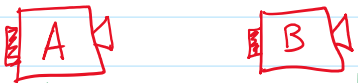
Thursday, August 29, 2024 11:21 AM

The Toolbox.

DEF: Investigate algorithm efficiency with respect to time and space.

Analytical not empirical

E.g.



¿which one is better?

Analytical:
we will represent performance as a mathematical function

R( ): input: size of the program's input data
output: number of "operations" performed by the program.

operations: ← + * = [] < >

$R_A(n)$ vs $R_B(n)$ n: size of the input data

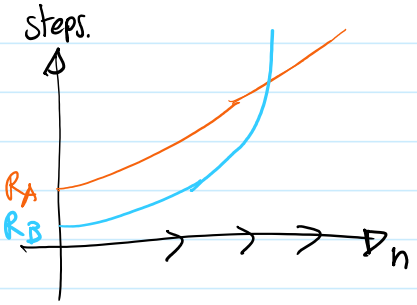
But?! which values of n to use?

- no specific values

we want to know what happens to the the Runtime function, as n becomes really large.

the Runtime function, as n becomes really large.

Graphs are deceitfull



- we are going to ignore constant factors

$R_{Bob}(n) = 3n^2$ $\quad$ $R_{Larry}(n) = 9n^2$

for our purposes $R_{Bob} = R_{Larry}$.

- we say that we care about a function's "rate-of-growth" or "order of growth"
what happens as $n \to \infty$

- But!?
there are many inputs of size n.
which one?

☆ - worst-input :- assume that the input takes the most operations.

- best-input :- optimistic underestimate.

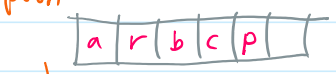
- all inputs and use the average

- amortized cost over all inputs

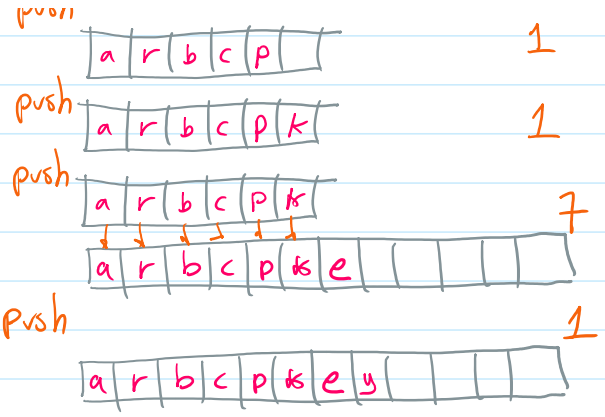
EG: (Amortized cost analysis)
Army Stack.

number of insertions

push



1



- ASYMPTOTIC NOTATION

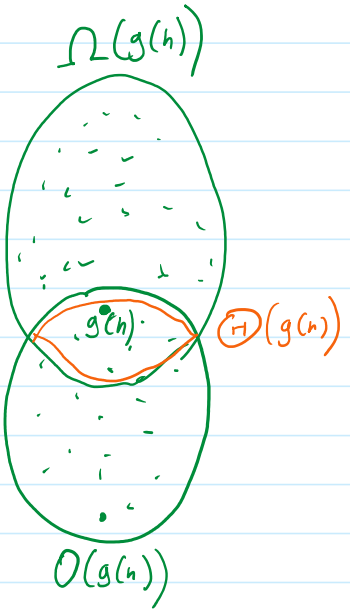
informal definitions:

if $g(n)$ is a function

$O(g(n))$ is the set of all functions with lower or the same "rate of growth"

$\Omega(g(n))$ is the set of all functions with a greater or the same "rate of growth"

$\Theta(g(n))$ is the set of all functions with the same rate of growth



Note: "rate of growth" ignores constant factors

E.G. functions in $O(n^2)$

- $n \in O(n^2)$
- $3n^2 \in O(n^2)$
- $100n^2 + 27n + 3 \in O(n^2)$

more:

- $n^3 \in \Omega(n^2)$
- $3n^2 \in \Omega(n^2)$
- $n! \in \Omega(n^2)$
- $3n^2 \in \Theta(n^2)$

DEF:

we say that $f(n) \in O(g(n))$ if there exists constants C and n_0 such that for all $n > n_0$

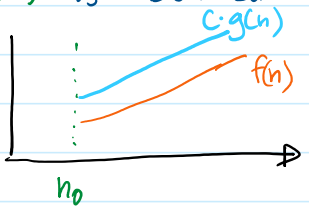
there exists constants C and n_0 such that for all $n > n_0$

$$f(n) \leq C \cdot g(n)$$

- no restriction on C and n_0
- C is what allows us to ignore constant factors.

In English:

$f(n)$ is bounded from above by a multiple of $g(n)$



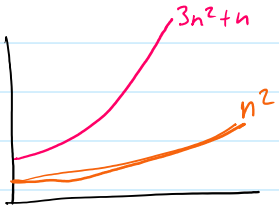
E.G. $n^2 \in O(3n^2 + n)$

need to show that: $n^2 \leq C \cdot (3n^2 + n)$ for n greater than some n_0

$C = 1$
$n_0 = 1$

$n^2 \leq 3n^2 + n$ for $n > 1$

$n^2 \leq 3n^2$ ☑



E.G. $3n^2 + n \in O(n^2)$

need to show that $3n^2 + n \leq C \cdot (n^2)$ for n greater than some n_0

$C = 4$
$n_0 = 1$

$3n^2 + n \leq 4 \cdot n^2$
$3n^2 + n \leq 3n^2 + 1n^2$
$n \leq n^2$ ☑

DEF —

we say that $f(n) \in \Omega(g(n))$ if there exists constants C and n_0 such that for every $n > n_0$

$$f(n) \geq C \cdot (g(n))$$

$$f(n) \geq C \cdot (g(n))$$

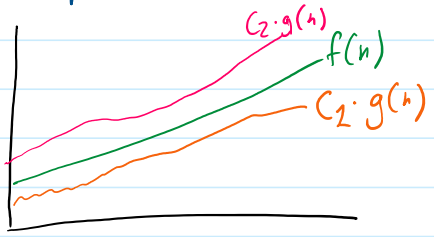
In English: $f(n)$ is bounded from below by a multiple of $(g(n))$

DEF –

we say that $f(n)$ is $\Theta(g(n))$ if there exists constants C_1, C_2 and n_0 such that for every $n > n_0$

$$C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n)$$

In English: $f(n)$ is bounded from above and below by multiples of $f(n)$

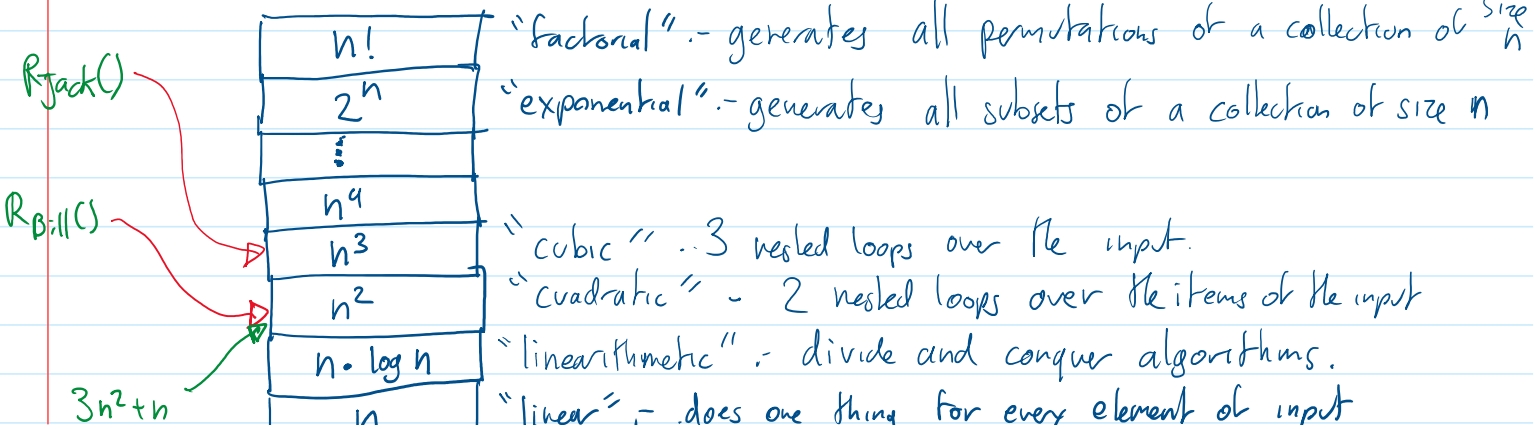


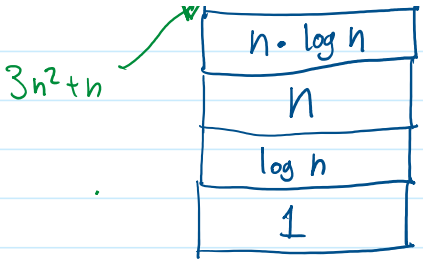
Note: if $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$ then $f(n) \in \Theta(g(n))$

- BASIC EFFICIENCY CLASSES:

Comparing functions:

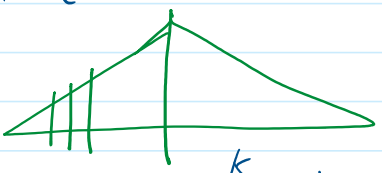
$$R_{Jack}(n) = 5n^3 + 2n^2 \qquad R_{Bill}(n) = \frac{1}{2}n^2 + 3$$





"linearithmetic" :- divide and conquer algorithms.
"linear" :- does one thing for every element of input
logarithmic :- split input in half and look at one part.
• "constants" - program without loops

Eg.- logarithmic



n = number of elements.

$2^k = n \qquad k = \log_2 n$

- MORE MATH : USEFUL PROPERTIES OF ORDER-OF-GROWTH

What if you need to run Jack followed by Bill

$$R_{JB} = R_{Jack}(n) + R_{Bill}(n)$$

- If you have functions $f_1(n), f_2(n), g_1(n), g_2(n)$ and you know that $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$

then

$$f_1(n) + f_2(n) \in O\left(\max\{g_1(n), g_2(n)\}\right)$$

So

$$R_{JB} = R_{Jack}(n) + R_{Bill}(n) \in O\left(\max\{n^3, n^2\}\right)$$

$$R_{JB} \in O(n^3)$$

- A polynomial of degree k is $O(n^k)$

- Limits and the order-of-growth.

$f(n)$ $\begin{cases} 0 \\ \infty \end{cases}$:- $f(n)$ has a smaller order-of-growth than $g(n)$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \begin{cases} 0 : f(n) \text{ has a smaller order-of-growth than } g(n) \\ C : f(n) \text{ has the same order-of-growth as } g(n) \\ \infty : f(n) \text{ has a larger order-of-growth than } g(n) \end{cases}$$

these allows us to use convenient results from calculus:

- L'Hopital rule:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)}$$

- Stirling's formula.

$$n! \approx \sqrt{2\cdot\pi\cdot n}\cdot\left(\frac{n}{e}\right)^n \quad \text{for large values of } n.$$

—EOF—