Cs 5050

04 01 2014

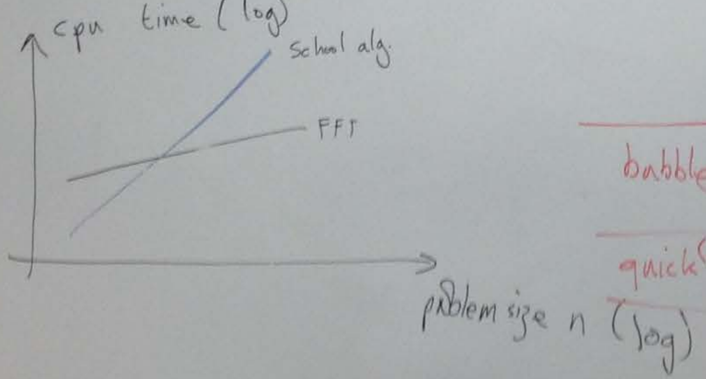
0

How fast / good Algorithm?

which algorithm is better? ← Depends
+ on problem size
+ data distribution

Method 1

Empirical studies



$f(n) = C_{FFT}\, n \log n$

constant (like 1000)

$t = C_{SA}\, n^2$

$\log t = 2n + \log C_{SA}$

example sorting

	almost sorted	random
bubble	Faster	slower
quick (pivot first)	slower	Fast

(1) Method 2

mathematical analysis → counting "steps"

recursive

recurrance relation

$$f(n) = 2f(n/2) + n$$

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

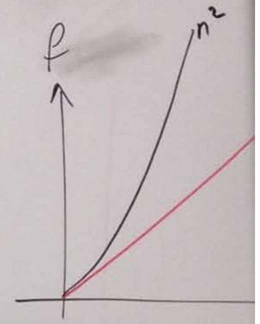
FFT

iterative

Sumation expression

$$f(n) = \sum_{i=1}^{n} \sum_{j=i}^{n} 1 \approx n^2$$

School algorithm



$n \log n = n^2$

(2) Abstraction counting

ignore + constants

+ low order terms

FFT is $O(n \log n)$

O read as "order"

School alg. is $O(n^2)$

3

$f(n)$ is $O(g(n))$

real constant $c > 0$

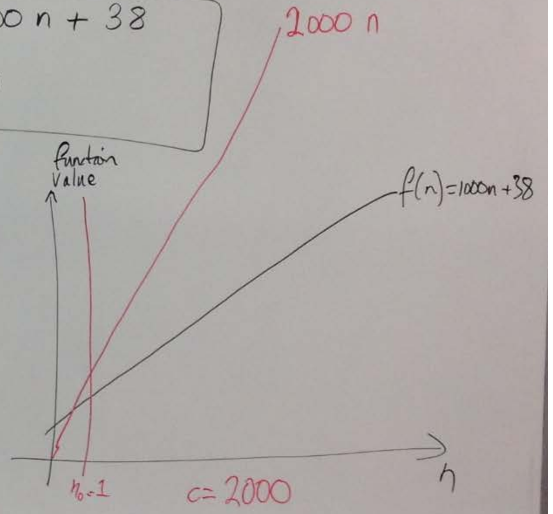
integer constant $n_0 \geq 1$

for all $n \geq n_0$ $\quad f(n) \leq c\, g(n)$

$f(n)$ is order $g(n)$

ex. $\boxed{\begin{aligned} f(n) &= 1238\, n \log_2 n \\ g(n) &= n \log n \end{aligned}}$

ex. $\boxed{\begin{aligned} f(n) &= 1000\, n + 38 \\ g(n) &= n \end{aligned}}$



4

$1000n + 88$ is $O(n)$

is order n

$\boxed{1.278n^2 + 0.2n + 0.01 \text{ is } O(n^2)}$

$n_0 = 2$ $c = 10$

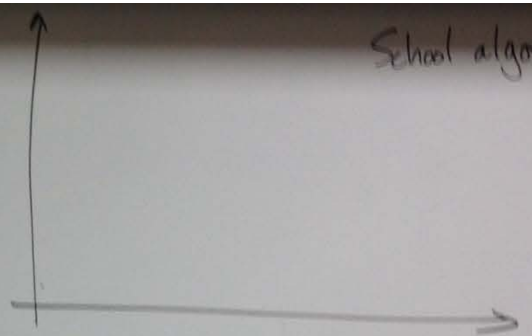
$\boxed{0.2n + 0.01 < 8.722n^2}$

$1.278n^2 + 0.2n + 0.01 < 10n^2$

for all $n \geq 2$

$1.278n^2 + 0.2n + 0.01$ is $O(n^3)$ $O(2^n)$

School algorithm.



cpu = $1.278n^2 + 0.2n + 0.01$ μs

real equation from the graph of the empirical study

(5) what does Order of $O(\ )$ mean?

+ ignore constants / lower order terms

how will the algorithm grow in time as n gets "very big"

asymptotic analysis

$O(n \log n)$ algorithm will be faster than $O(n^2)$ algorithm eventually

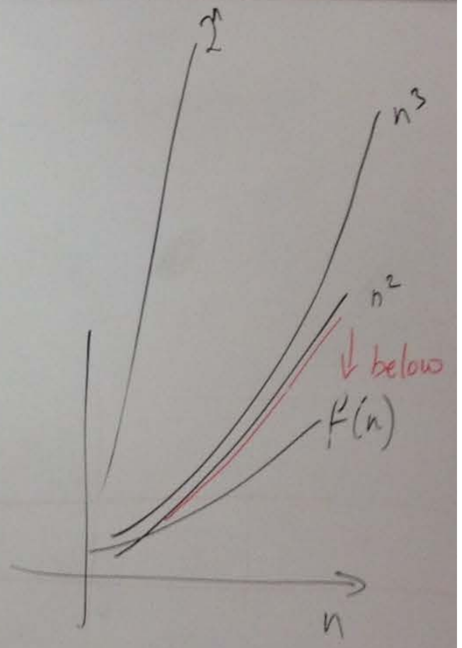
6 $f(n)$ is $O(g(n))$

$g(n)$ is an $\boxed{\text{upper bound}}$ on $f(n)$

"no worse than"

$f(n)$ is bound from above by $g(n)$

want is a $\boxed{\text{tight}}$ bound



7 Nim DP $n\,S$ — constant

rec. 2^n

Sorting n^2 , $n \log n$

$n^{1.579}$

$O(\log n) < O(n) < O(n \log n) < O(n^{1.579}) < O(n^2) < O(2^n) \xrightarrow{\ n\ }$

$/n \log n$