Col100 : Lec 6

Order of growth

- m/c independent measure of the resources reqd. by a computational process
- If n is a measure of the size of the problem then R(n) be the function that measures the resources required
- R(n) has an order of growth $O(f(n))$ if $\exists\ k, n_0$ s.t. $R(n) \leq k f(n)$ for $n \geq n_0$

Advantage of big-O function

- hides m/c specific constants
- fairly precise indication of how algorithm scales as we increase the size of i/p
- tells us which of the several competing algorithms will win in the long run

It is an Asymptotic analysis

Q: Why not just take the first derivative of these functions to get order of growth

A: Not continuously differentiable, functions depend on IN as i/ps → discrete

formal defⁿ -

for any function $g: \mathbb{N} \longrightarrow \mathbb{R}$ that is asymptotically non-negative, $O(g(n))$ is the class of functions

s.t.

$$O(g(n)) = \{ f(n) \mid \exists\, c \in \mathbb{R}, n_0 > 0 : \forall\, n \geq n_0 : 0 \leq f(n) \leq c \cdot g(n) \}$$

for any $f(n) \in O(g(n))$

we write

$$f(n) = O(g(n))$$

Abuse of notation

Examples

$n = O(n)$

$n = O(n^2)$

$n = O(2^n)$

but

$n \neq O(\log_2 n)$

$n \neq O(1)$

Some facts

- if $f(n) = O(g(n))$ then

  $O(f(n)) \subseteq O(g(n))$

- if $f(n) = O(g(n))$

  and $g(n) = O(f(n))$

  then $\boxed{f(n) = \Theta(g(n))}$

$\exists\, c_1, c_2, n_0$ s.t. $n \geq n_0 > 0$

$c_1 g(n) \leq f(n) \leq c_2 g(n)$

- if $f(n) = O(g(n))$

  then $af(n) = O(g(n))$

  for any real constant a

- if $d(n) = O(f(n))$ and
  $e(n) = O(g(n))$ then

  (a) $d(n) \cdot e(n) = O(f(n) \cdot g(n))$

  (b) $d(n) + e(n) = O(f(n) + g(n))$

- if $d(n) = O(f(n))$ and $f(n) = O(g(n))$
  then $d(n) = O(g(n))$

Eg

$$\text{fastpower}(x, n) = \begin{cases} 1 & \text{if } n = 0 \\ \text{fastpower}(x^2, n \text{ div } 2) & \text{if } n > 0 \text{ and } n \bmod 2 = 0 \\ x \cdot \text{fastpower}(x^2, n \text{ div } 2) & \text{otherwise} \end{cases}$$

Worst Case Computation

- when the exponent always remains odd till it reaches a value 1 at some k^{th} - step

fast power (x, n)

Step 0 $= x \cdot$ fastpower $(x^2,\ n \text{ div } 2)$

Step 1 $= x \cdot x^2$ fastpower $(x^4,\ (n \text{ div } 2) \text{ div } 2)$

⋮

Step K $= \underbrace{x_0\ x_1 \ldots\ x_{K-1}}_{K \text{ multiplications}}$

where $x_0 = x$ $\quad n_0 = n$

$x_{i+1} = x_i^2$ $\quad n_{i+1} = n_i \text{ div } 2$

$\forall i \geq 1$

$O(K)$ $\quad K = \log_2 n$

why?

$2^{K-1} \leq n < 2^K$

i) $n_0 = 2^K - 1$

then $n_i = n_{i-1} \text{ div } 2$

$= 2^{K-i} - 1$

for all i,

$0 < i \leq K$

- prove by induction

Space requirement

$= O(K)$

$= O(\log_2 n)$

Can we improve the space complexity?

$$fastpower2(x,n) = \begin{cases} 1 & \text{if } n=0 \\ fastpower2(x,n,1) & \text{if } n>0 \end{cases}$$

fastpower_tr (y, m, p)

$$= \begin{cases} p & \text{if } m = 0 \\ \text{fastpower}(y^2, m \text{ div } 2, p) & \text{if } m \bmod 2 = 0 \\ \text{fastpower}(y^2, m \text{ div } 2, p \cdot y) & \text{otherwise} \end{cases}$$

Correctness?

$\forall\ y, p > 0$ and $m \geq 0$

fastpower_tr $(y, m, p) = p \cdot y^m$ | induct on m

Basis : fastpower_tr $(y, 0, p) = p = p \cdot y^0$

I.S: two cases

Say $m = 2j > 0$

I.H holds for $0 \leq k < m$

fastpower $(y, 2j, p)$

$=$ fastpower (y^2, j, p)

$\because j < m$, IH applies

fastpower_tr $(y^2, j, p) = p \cdot (y^2)^j$

$= p \cdot y^{2j}$

$= p \cdot y^m$

Similarly show the odd step

Analysis of fastpower-tr

- fastpower-tr (x_0, n_0, p_0)

  fastpower-tr (x_1, n_1, p_1)

  ⋮

  fastpower-tr $(x_k, 0, p_k)$

  $= p_k$ each of

Worst case we have $n_0, \ldots n_{k-1} = 1$ is an odd number

$$T(\text{fastpower-tr}(x_0, n_0, p_0)) = 2 + T(\text{fastpower}(x_1, n_1, p_1))$$

$$n_0 = 2^k - 1$$
$$x_{i+1} = x_i^2$$
$$n_{i+1} = n_i \text{ div } 2$$

$$p_i = \prod_{j=0}^{i-1} x_j \quad 0 \le i < k$$

Notice that

$$n_K = 0, \quad p_0 = 1$$

Hence

$$T(\text{fastpower\_tr}(x_0, n_0, p_0))$$

$$= 2k + T(\text{fastpower}(x_K, 0, p_K))$$

$$\cong O(k)$$

$$= O(\log_2 n)$$

But space req.

$$\int(\text{fastpower\_tr}(x_0, n_0, p_0)) = \int(\text{fastpower\_tr}(x_1, n_1, p_1))$$

$$= \ldots$$

$$= O(1)$$

Eg: Integer Square Root