OH Today: 1:15pm - 2:15pm

Requesting extensions

Running Time of Algorithms

- Implement the alg & check it against many inputs.

Drawbacks:

1. We need to implement the alg.
2. We will not be able to check against all inputs.
3. To compare two alg, we will need to implement them using the same software & hardware environments.

Analytical analysis: takes care of all inputs and we can compare algorithms independently.
of the the software & hardware environment.

\[ T(n) \]

\[ \text{RAM model of computation.} \]

- all high-level operations take constant time.
  
  arithmetic operations, comparisons, return stmt, 
  fn. call, indir. into an array, ...

- high level oper.
  
  \[ \exists c \text{ s.t. } f(n) \leq cn \text{ for large } n \]

- all memory accesses take unit time.

Worst case analysis of the algo.

Order of growth (Asymptotic analysis)

\[ n \to \infty \]

- ignore const.
- drop lower order terms.

\[ O(g(n)) = \{ f(n) \mid \exists \text{ positive const } c \text{ and } n_0 \text{ s.t. } 0 \leq f(n) \leq c g(n), \forall n \geq n_0 \} \]

\[ \Omega(g(n)) = \{ f(n) \mid \exists \text{ positive const } c \text{ and } n_0 \text{ s.t. } f(n) \geq c g(n) > 0, \forall n \geq n_0 \} \]

lower-bounded

\[ f(n) \]
\[ \Theta(g(n)) = \left\{ f(n) \mid \exists \text{ positive const } c_1, c_2 \text{ & } n_0 \text{ s.t.} \right. \]
\[ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n), \quad n \geq n_0 \left. \right\} \]

\[ f(n) = \Theta(g(n)) \quad \text{iff} \quad f(n) = O(g(n)) \text{ and } f(n) = \Omega(g(n)). \]
Ex. Prove that $f_{n-2} = \Theta(n)$

Proof: $f_{n-2} \leq f_n \quad \forall \, n \geq 1$

$c_2 = 7, \quad n_0 = 1$

To prove the lower bound, we must show that $f_{n-2} \geq c_1, \quad \forall \, n \geq n_0$

i.e. $(f - c_1) n \geq 2$

Let $c_1 = 3$, and let $n_0 = 2$.

$\therefore \quad f_{n-2} = \Theta(n)$.

$c_1 = 3, \quad c_2 = 7, \quad n_0 = 2$.

Ex: Prove that $10n^3 + 55n \ln n + 23 = O(n^3)$.

Proof: $10n^3 + 55n \ln n + 23 \leq 10n^3 + 55n^3 + 23n^3, \forall \, n \geq 1$

$= 88n^3$
Given: \( c = 88 \quad n_0 = 1 \),
\[ 8000 \]

Ex: Prove that \( 3^{100} = O(1) \).

**Proof**: \( 3^{100} \leq 3^{100} \cdot 1 \), \( \forall n > 1 \)

\[ \therefore \text{ claim follows.} \]

Ex: Prove that \( \frac{n^2}{8} - 50n = \Theta(n^2) \)

**Proof**: \( \frac{n^2}{8} - 50n \leq n^2 \), \( \forall n > 400 \)

\[ \geq 0 \]

\[ \frac{n^2}{8} - 50n \geq 0 \]

\[ \therefore n \geq 400 \]

\[ c_2 = 1, \quad n_0 = 400 \]

\[ \text{To prove that } n^2 - 50n = \Omega(n^2), \text{ we need to} \]
Show that \( \frac{n^2}{8} \geq 50n \), \( \forall n \geq n_0 \).

i.e., \( \frac{n}{8} \geq 50 \), \( \forall n \geq n_0 \).

\[ \therefore \quad n \left( \frac{1}{8} - C_1 \right) \geq 50 \]

Let \( C_1 = \frac{1}{16} \). Plugging this value, we get

\[ n \cdot \frac{1}{16} \geq 50 \]

\[ n \geq 800 \]

\[ \therefore \quad n_0 = 800 \quad \& \quad C_1 = \frac{1}{16}. \]

Thus \( C_1 = \frac{1}{16}, \quad C_2 = 1, \quad n_0 = \max \{400, 800\} = 800 \).
Ex: Prove that $\log n = O(n)$.

Proof: We will prove the claim using induction on $n$. We will prove that $\log n \leq n$, for all $n \geq 1$.

IH: Assume that $\log k \leq k$, for some $k \geq 1$.

BC: $\log 1 = 0 \leq 1$  

IS: Want to prove that

$\log (k+1) \leq k+1$

LHS = $\log k + \log 1$

$\leq k + 0$

$< k+1$  

$\checkmark$. 


\[ \text{LHS} = \lg (k+1) \]
\[ \leq \lg (k+k) \]
\[ = \lg (2k) \]
\[ = \lg 2 \cdot \lg k \]
\[ \leq 1 \cdot k \]
\[ \leq k+1. \]

Ex: Prove that \( 3^{100} = O(2^n) \).

Proof: We will prove that
\[ 3^{100} \leq c \cdot 2^n, \quad \forall n \geq n_0. \]

IH: Assume that \( 3^{100} \leq (3 \cdot 100)^k \), for
Some $k \geq 100 \cdot 200$.

**TH:** Want to prove that

$$3 \cdot (k+1)^{100} \leq (3 \cdot 100)^2 \cdot k^{1+}$$

**LHS:**

$$= 3 \cdot (k+1)^{100}$$

$$= 3 \cdot \left[ (\begin{array}{c} 100 \\ 0 \end{array}) \cdot k^{99} + (\begin{array}{c} 100 \\ 1 \end{array}) \cdot k^{98} + \cdots + (\begin{array}{c} 100 \\ 99 \end{array}) \cdot k^0 \right]$$

$$\leq 3 \left[ k + 100 \cdot k + 100^2 \cdot k + \cdots + 100^{99} \cdot k^0 \right]$$

$$\leq 3 \cdot \left[ \frac{100}{k} + \left( \frac{100}{k} \right)^2 + \cdots + \left( \frac{100}{k} \right)^{100} \right]$$

By IH

$$\leq 3 \cdot 100 \cdot 2^k \left( \begin{array}{c} 100 \\ 2 \end{array} \right)$$

$$\leq 3 \cdot 100 \cdot 2^k \left( \begin{array}{c} 100 \\ 2 \end{array} \right)$$

$$\leq (\frac{1}{2})^0 + (\frac{1}{2})^1 + \cdots + (\frac{1}{2})^{100}$$

$$\leq (\frac{1}{2})^0 = 1$$
BC: \( n = 200 \).

\[
\frac{3 \cdot 200}{100} \leq \left( \frac{3 \cdot 100}{100} \right) \cdot 2^{200}
\]

\[
= \left( 3 \cdot 2 \right) \cdot 100.
\]

Ex: Prove that \( f_{n-2} \) is not \( \Omega \left( n^{10} \right) \).

Proof: Assume that \( f_{n-2} = \Omega \left( n^{10} \right) \)

\[
\therefore f_{n-2} \geq cn^{10}, \quad \forall n \geq n_0.
\]

\[
f_n \geq c \cdot n^{10}, \quad \forall n \geq n_0
\]

\[
f \geq \left[ c \cdot n \right], \quad \forall n \geq n_0.
\]

This is clearly not true because
as soon as \( n > \frac{3}{2} \left( \frac{1}{c} \right)^{\frac{1}{4}} \), RHS is bigger than LHS.

c. \( (3 \left( \frac{1}{c} \right)^{\frac{1}{4}})^9 \)

\[ 3^9 \]

\[ \text{little-o, o} \]

\[ f(n) = O(g(n)), \text{ but } f(n) \neq \Theta(g(n)). \]

\[ n^2 = O(n^3), \text{ but } n^2 \neq \Theta(n^3) \text{ then} \]

\[ n^2 = o(n^3). \]

\[ \text{little-omega, o} \]

\[ f(n) = \Omega(g(n)), \text{ but } f(n) \neq \Theta(g(n)). \]

\[ f(n) = n^2, g(n) = n, \text{ but } f(n) = \omega(g(n)). \]
\[
\lim_{n \to \infty} \frac{f(n)}{g(n)} = \begin{cases} 
0, & f(n) = o(g(n)) \\
\infty, & f(n) = \omega(g(n)) \\
c, & f(n) = \Theta(g(n)).
\end{cases}
\]

Tight bound = \Theta-bound.

Properties:

1. \( f(n) = O(g(n)) \) and \( g(n) = O(h(n)) \) then \( f(n) = O(h(n)) \).

2. Same for \( \omega \).

3. Same for \( \Theta \).

4. \( f(n) = O(h(n)) \) and \( g(n) = O(h(n)) \) then \( f(n) + g(n) = O(h(n)) \).
Polynomial function of degree $d$

\[ f = a_0 + a_1 \cdot n + a_2 \cdot n^2 + \ldots + a_d \cdot n^d, \ \text{where } a_d \neq 0. \]

\[ f = O(n^d). \]

For every $b > 1$ and any real nos $x, y > 0$, we have:

\[(\log_b n)^x = O(n^y).\]

poly-log

\[ (\log n)^{100} = O(n^{0.1}) \]

For every $r > 1$ and every $d > 0$, we have

\[ n^d = O(r^n). \]

\[ (\log n)^3 \]
Ex: Prove that \( \frac{\log n}{n} = o \left( \frac{\log^2 n}{n} \right) \).

We are given

\[
\frac{10}{n} \cdot \frac{\log^2 n}{n} = o \left( \frac{\log^2 n}{n} \right).
\]

From what we discussed,

\[
\lim_{n \to \infty} \frac{10}{\log^2 n} = 0
\]

Let \( x = \log n \)

\[
10 = 0 \left( \frac{\log^2 n}{n} \right).
\]

We have a processor that performs a million high-level instructions.

\[
\log n, \log \log n, n, n^2, 2^n, n^k, 2^n, \log^2 n
\]