I. Algorithm Analysis

- Algorithm analysis is to identify the time category (or order of growth) of an algorithm
- Eight most common categories
  - $1, \log n, n, n(\log n), n^2, n^3, 2^n, n!$

Most Popular

Three Notations

O (Big Oh) — O

Q (Big Theta) — Θ

W (Big Omega) — Ω

O: Best & Worst Case differ
Θ: Always same $t(n)$
Ω: Lower Bound

II. Big-Oh (O) Notation

- Ex: Algorithm SequentialAdd (A[0...n-1])

```
sum ← 0
i ← 0
while (i<n) do
    sum ← sum + A[i]
    i ← i+1
return sum
```

Rules To Simplify T(n)

1. Eliminate low order terms
   $T(n) = 4*n + 5 \rightarrow 4*n$
2. Eliminate Constant Coefficients
   $T(n) = 4*n \rightarrow n \rightarrow O(n)$

- Big O notation denotes UPPER BOUND

$t(n) \leq c*g(n)$

$O(n\log n)$
$n\log n$ or faster

- Big-O notation with Basic Operation
  - It's difficult to calculate the total number of all executions of all operations
    - To simplify the calculation, we consider only the number of executions of the basic operation
- Big-O formal definition:
  - An algorithm's running time $t(n)$ is said to be in $O(g(n))$, denoted $t(n) \in O(g(n))$, if there exist constants c & $n0$ that satisfy the following condition

$g(n)$

One of ϕ categories

$$t(n) \text{ £ } c * g(n) \text{ for all } n^3 n_0$$

III. Big Omega Notation (W) (Ω)

- An algorithm's running time $t(n)$ is said to be in $W(g(n))$, denoted $t(n) \in W(g(n))$. If there exist constants c & $n0$ that satisfy the following condition

$$t(n)^3 c * g(n) \text{ for all } n^3 n_0$$

- Denotes LOWER BOUND

$\Omega(n * \log n)$

$n\log n$ or slower

$$t(n) \geq c * g(n)$$

IV. Big Theta Notation (Q) (Θ)

- An algorithm's running time t(n) is said to be in Q(g(n)), denoted t(n) $\in$ Q(g(n)) if there exist constants c1, c2, & n0 that satisfy the following condition

$$c1 * g(n) \leq t(n) \leq c2 * g(n) \text{ for all } n^3 n_0$$

Scratch: O Ω

$t(n) = n^2+3n+1 \in O(n)$?

$t(n) = n^2+3n+1 \leq C * n$ for all $n \geq n_0$

Impossible, because n^2 will increase faster than n

if $t(n) = n$, can we say $t(n) \in O(n)$? Yes

if $t(n) = n$, can we say $t(n) \in O(n^2)$? Yes

Meets upper bound (or faster)

if $t(n) = 2 \times n + 5$ can we say $t(n) \in \Omega(n)$? Yes

Can we say $t(n) \in \Omega(n^2)$

$t(n) = 2 \cdot n + 5 \geq c * n^2$ for all $n \geq n_0$

not possible

O = or faster

Ω = or slower

$t(n) \geq c * (\Omega \cdot n \log n)$ for all $n \geq n_0$

$2n+5 \geq c * (\Omega \cdot n \log n)$ for all $n \geq n_0$

not possible

Scratch:

Theta ex 2

$$c_1 \cdot n \le 2 * n + 3 \le c_2 \cdot n$$

$c_1 : 1$ $\quad n \le 2n+3 \le 5n$

$c_2 : 5$ $\quad$ True

5 minutes

0 0 0 0 0

1 1 1

2 2 1

3 2 2

4 1 1

5 2 2