COL202: Discrete Mathematical Structures

Ragesh Jaiswal, CSE, IIT Delhi

Algorithms

Definition (Big-O)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ (or $f(n) = O(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \leq c \cdot g(n)$$

- Another short way of saying that $f(n) = O(g(n))$ is "$f(n)$ is **order of** $g(n)$".
- Show that: $8n + 5 = O(n)$.

Definition (Big-O)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ (or $f(n) = O(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \leq c \cdot g(n)$$

- Another short way of saying that $f(n) = O(g(n))$ is "$f(n)$ is **order of** $g(n)$".
- Show that: $8n + 5 = O(n)$.
  - For constants $c = 13$ and $n_0 = 1$,we show that $\forall n \geq n_0, 8n + 5 \leq 13 \cdot n$. So, by definition of big-O, $8n + 5 = O(n)$.

Definition (Big-O)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ (or $f(n) = O(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \leq c \cdot g(n)$$

- Another short way of saying that $f(n) = O(g(n))$ is "$f(n)$ is **order of** $g(n)$".
- $g(n)$ may be interpreted as an upper bound on $f(n)$.
- Show that: $8n + 5 = O(n)$.
- Is this true $8n + 5 = O(n^2)$?

Definition (Big-O)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ (or $f(n) = O(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \leq c \cdot g(n)$$

- Another short way of saying that $f(n) = O(g(n))$ is "$f(n)$ is **order of** $g(n)$".
- $g(n)$ may be interpreted as an upper bound on $f(n)$.
- Show that: $8n + 5 = O(n)$.
- Is this true $8n + 5 = O(n^2)$? Yes
- $g(n)$ may be interpreted as an *upper bound* on $f(n)$.
- How do we capture *lower bound*?

Definition (Big-Omega)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $\Omega(g(n))$ (or $f(n) = \Omega(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \geq c \cdot g(n)$$

Definition (Big-Omega)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $\Omega(g(n))$ (or $f(n) = \Omega(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \geq c \cdot g(n)$$

- Show that: $f(n) = \Omega(g(n))$ iff $g(n) = O(f(n))$.

Definition (Big-O)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ (or $f(n) = O(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \leq c \cdot g(n)$$

Definition (Big-Omega)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $\Omega(g(n))$ (or $f(n) = \Omega(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \geq c \cdot g(n)$$

- How do we say that $g(n)$ is both an upper bound and lower bound for a function $f(n)$? In other words, $g(n)$ is a **tight bound** on $f(n)$.

Definition (Big-O)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ (or $f(n) = O(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \leq c \cdot g(n)$$

Definition (Big-Omega)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $\Omega(g(n))$ (or $f(n) = \Omega(g(n))$ in short) **iff** there is a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that:

$$\forall n \geq n_0, f(n) \geq c \cdot g(n)$$

Definition (Big-Theta)

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $\Theta(g(n))$ (or $f(n) = \Theta(g(n))$) **iff** $f(n)$ is $O(g(n))$ and $f(n)$ is $\Omega(g(n))$.

- Question: Show that $3n \log n + 4n + 5 \log n$ is $\Theta(n \log n)$.

- Growth rates:
  - Arrange the following functions in ascending order of growth rate:
    - n
    - $2^{\sqrt{\log n}}$
    - $n^{\log n}$
    - $2^{\log n}$
    - $n/\log n$
    - n^n

Introduction

- Given two algorithms A1 and A2 for a problem, how do we decide which one runs faster?
- What we need is a platform independent way of comparing algorithms.
- Solution: Do an asymptotic worst-case analysis recording the running time using Big-(O, Ω, Θ) notation.

Introduction

- How do we describe an algorithm?
  - Using a pseudocode.
- What are the desirable features of an algorithm?
  1. It should be correct.
     - We use proof of correctness to argue correctness.
  2. It should run fast.
     - We do an asymptotic worst-case analysis noting the running time in Big-(O, Ω, Θ) notation and use it to compare algorithms.

Example

`FindPositiveSum`(A, n)	
- $sum \leftarrow 0$	$O(1)$
- For $i = 1$ to n	$O(n)$
- if $(A[i] > 0) sum \leftarrow sum + A[i]$	$O(n)$
- return(sum)	O(1)
	Total: $O(n)$

Algorithms

- Algorithms: Does there exist a problem that cannot be solved by any algorithm?

End