Algorithms Design and Analysis [ETCS-301]

Dr. A K Yadav
Amity School of Engineering and Technology
(affiliated to GGSIPU, Delhi)
akyadav1@amity.edu
akyadav@akyadav.in
www.akyadav.in
+91 9911375598

July 24, 2019

Introduction to algorithms I

- What is an algorithm?
  - An algorithm is a set of rules for carrying out calculation either by hand or on a machine.
  - An algorithm is a sequence of computational steps that transform the input into output.
  - An algorithm is a sequence of operations performed on data that have to be organized in data structures.
  - A finite set of instructions that specify a sequence of operations to be carried out in order to solve a specific problem or class of problems.
  - An algorithm is an abstraction of a program to be executed on a physical machine.
  - *An algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.*

Introduction to algorithms II

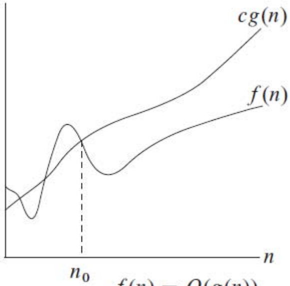
- Why do we study algorithms?
  - To make solution more faster.
  - To compare performance as a function of input size.
- Properties
  - Finiteness
  - Unambiguous
  - Sequence of execution
  - Input/output
  - Feasible
- Algoritm vs Program
  - Algorithm is independent of programming language and written in any natural language.
  - Implementation of any algorithm in a programming language is a program.

Introduction to algorithms III

- Design Techniques
  - Divide and Conquer (D&C)
  - Dynamic Programming (DP)
  - Greedy Approach
  - Branch and Bound
  - Backtracking
  - Randomized
- Random Access Machine (RAM)
  - Simply count primitive operations
  - Use pseudo code
  - Independent of programming language
  - Equal time for all operations

Asymptotic notations I

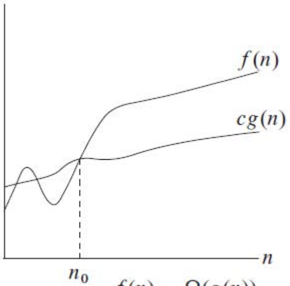
1. O - notation "Big O" : Asymptotic upper bound,
   $O(g(n)) = \{f(n) : \exists c, n_0 > 0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$
   $f(n) \in O(g(n))$
   $f(n) = O(g(n))$



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

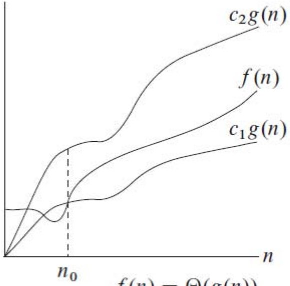
Asymptotic notations II

2. Ω - notation "Big omega" : Asymptotic lower bound,
$\Omega(g(n)) = \{f(n) : \exists c, n_0 > 0 \text{ such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0\}$
$f(n) \in \Omega(g(n))$
$f(n) = \Omega(g(n))$



$f(n) = \Omega(g(n))$

3. Θ - notation : Asymptotic tight bound,
$\Theta(g(n)) = \{f(n) : \exists c_1, c_2, n_0 > 0$ such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0\}$
$f(n) \in \Theta(g(n))$
$f(n) = \Theta(g(n))$



$f(n) = \Theta(g(n))$

Asymptotic notations IV

4. o - notation "small o": Asymptotic loose upper bound, $o(g(n)) = \{f(n) : \forall c > 0, \exists n_0$ such that $0 \leq f(n) < cg(n)$ for all $n \geq n_0\}$
5. ω - notation "small omega": Asymptotic loose lower bound, $\omega(g(n)) = \{f(n) : \forall c > 0, \exists n_0$ such that $0 \leq cg(n) < f(n)$ for all $n \geq n_0\}$

Benefits of Asymptotic Notations:

- Simple representation of algorithm efficiency.
- Easy comparison of performance of algorithms.

Relationship between notations I

- $f(n) = \Theta(g(n)) \Rightarrow f(n) = O(g(n))$
- $f(n) = \Theta(g(n)) \Rightarrow f(n) = \Omega(g(n))$
- $f(n) = O(g(n)) \& f(n) = \Omega(g(n)) \Rightarrow f(n) = \Theta(g(n))$
- $f(n) = O(g(n))$ may or may be tight upper bound but $f(n) = o(g(n))$ is always upper loose bound and $o(g(n)) \in O(g(n))$ and $o(g(n)) \subset O(g(n))$
- $f(n) = \Omega(g(n))$ may or may be tight lower bound but $f(n) = \omega(g(n))$ is always lower loose bound and $\omega(g(n)) \in \Omega(g(n))$ and $\omega(g(n)) \subset \Omega(g(n))$
- $$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \begin{cases} 0 & f(n) = o(g(n)) \\ \infty & f(n) = \omega(g(n)) \\ c & 0 < c < \infty, f(n) = \Theta(g(n)) \end{cases}$$

Relationship between notations II

- Let $f(n) = \sum_{i=0}^{d} a_i n^i$ where $a_d > 0$, is a degree-d polynomial in n and k is a constant then:
  - if $k \geq d$, then $f(n) = O(n^k)$.
  - if $k \leq d$, then $f(n) = \Omega(n^k)$.
  - if $k = d$, then $f(n) = \Theta(n^k)$.
  - if $k > d$, then $f(n) = o(n^k)$.
  - if $k < d$, then $f(n) = \omega(n^k)$.

Thank you

Please send your feedback or any queries to akyadav1@amity.edu, akyadav@akyadav.in or contact me on +91 9911375598