Asymptotic Notations

Acknowledgements: Dr. Saleena N, CSED

Rate of Growth or Order of Growth

- Rate/Order of growth – Considering the leading term of a formula
- Ignoring the lower order terms - insignificant for large values of n
- Ignoring leading term’s constant coefficient – constant factors are less significant

Rate of Growth or Order of Growth

- $T(n) = an^2 + bn + c$
- we say $T(n)$ is $\Theta(n^2)$ ("theta of *n*-squared")
  - simplifying abstraction
  - consider only the leading term of a formula
  - lower order terms- relatively insignificant for large values of n
  - ignore the leading term's constant coefficients

Asymptotic Efficiency

- input size is large enough, only the order of growth is relevant
- asymptotic efficiency - how the running time increases with the size of the input *in the limit*
- asymptotically more efficient - best choice for all but very small inputs

Asymptotic Notations

- Domain of functions - Set of Natural Numbers
  $\mathbb{N} = 0, 1, 2....$ (as given by CLRS)
  - $T(n)$ usually defined only on integer input sizes

- $T(n) = an^2 + bn + c$
  - $T(n)$ is $\Theta(n^2)$ ("theta of n-squared")
  - $T(n)$ is $O(n^2)$ ("Big - Oh of n-squared")
  - $T(n)$ is $\Omega(n^2)$ ("Omega of n-squared")

O notation (big-Oh)

- $T(n) = n^2 + 2n + 1$ for $n > 1$, $\quad T(1) = 4$
- $T(n) \leq 4n^2$, for $n \geq 1$
- $T(n) \leq cn^2$, for $n \geq n_0$ ($c=4$ and $n_0=1$)
- we say $T(n)$ is $O(n^2)$

O notation (big-Oh)

- $T(n) = n^2 + 2n + 1$ for $n > 1$, $T(1) = 4$
- How to get c and n_0 such that $T(n) \leq cn^2$, for $n \geq n_0$
  - c should be such that $n^2 + 2n + 1 \leq cn^2$
  - divide by n^2, $1 + \frac{2}{n} + \frac{1}{n^2} \leq c$
  - for $n \geq 1$, we can choose $c \geq 4$
- $T(n) \leq cn^2$, for $n \geq n_0$ (c=4 and n_0=1)
- we say $T(n)$ is $O(n^2)$
- $n^2 + 2n + 1$ is $O(n^2)$

O notation

- $T(n)$ is $O(n^2)$
  - There are positive constants c and n_0 such that $T(n) \leq cn^2$ for $n \geq n_0$

Some functions:

$T_1(n) = 5n^2$ $T_2(n) = n^2 + 2n$ $T_3(n) = n+5$

- $T_1(n) \leq 5n^2$ for $n \geq 1$
- $T_2(n) \leq 2n^2$ for $n \geq 2$
- $T_3(n) \leq n^2$ for $n \geq 3$

Generalizing…..

- *There exists positive constants $c=5$ and $n_0=1$ such that* $T_1(n) \le cn^2$ *for* $n \ge n_0$
- *There exists positive constants $c=2$ and $n_0=2$ such that* $T_2(n) \le cn^2$ *for* $n \ge n_0$
- *There exists positive constants $c=1$ and $n_0=3$ such that* $T_3(n) \le cn^2$ *for* $n \ge n_0$

There exists positive constants c and n_0 such that $f(n) \le cn^2$ ***for*** $n \ge n_0$

The set $O(n^2)$ *(read "big oh of n^2" or "oh of n^2")*

- *Set of all $f(n)$ such that there exists positive constants c and n_0 such that $f(n) \le cn^2$ for all $n \ge n_0$*

  Set is denoted by $O(n^2)$
  - $T_1(n) \in O(n^2)$ $T_2(n) \in O(n^2)$ $T_3(n) \in O(n^2)$
- **$O(n^2)$** is a set of functions.

$O(n^2) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \le f(n) \le cn^2$ for all $n \ge n_0 \}$

The set $O(n^2)$ – contd.

- Give some more functions that belong to the set $O(n^2)$.

$f_1(n) = 100n^2 + n + 5$

$f_2(n) = 6n + 3$

$f_3(n) = 10000n^2$

The set $O(n^3)$

- $O(n^3) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \leq f(n) \leq cn^3$ for all $n \geq n_0 \}$

- **Some of the elements of $O(n^3)$**

$f_4(n) = 100n^3 + 3n^2 + 2$

$f_5(n) = 6n+3$

$f(n)_6 = 10000n^2$

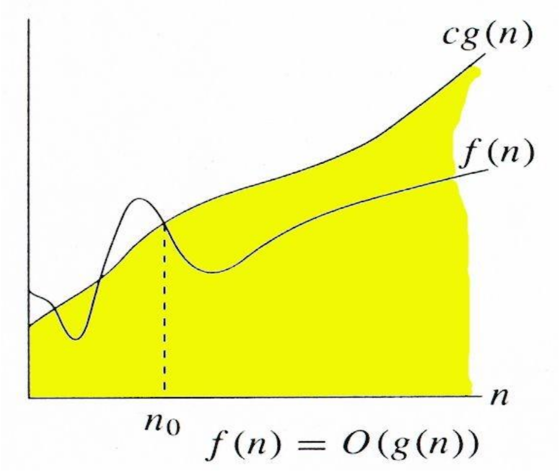
Generalizing….

- $O(n) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \le f(n) \le c\,n$ for all $n \ge n_0 \}$
- $O(n \lg n) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \le f(n) \le c\,n \lg n$ for all $n \ge n_0 \}$

$O(g(n)) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \le f(n) \le c.\,g(n)$ for all $n \ge n_0 \}$

O-notation gives an **upper bound for a function** to within a constant factor.

f(n)=O(g(n)), if there are positive constants ***c and n₀*** *such that to the right of* $\boldsymbol{n_0}$*, the value of* ***f(n) always lies on or below cg(n)***.



Source: http://www.cs.unc.edu/~plaisted/comp122/02-asymp.ppt

Going back ….

- **Insertion Sort**

  Worst Case Running time is $\mathbf{O(n^2)}$

- *Worst Case Running time,* $T_1(n) \leq cn^2$ ***for all values of*** $n \geq n_0$ *where c and* n_0 *are positive constants.*

Normally we write f (n) is O(g(n)) or f(n)=O(g(n)) to mean "f (n) is a member of O(g(n))"

Prove $T(n) = n^3+20n+1$ is $O(n^3)$

- by the Big-Oh definition, $T(n)$ is $O(n^3)$ if $T(n) \le c \cdot n^3$ for some $n \ge n_0$
- Find out c and n_0

Exercises

1. Is $2n + 10 \in O(n^2)$?

2. Is $n^3 \in O(n^2)$?

The set $\Omega(n)$ (*Read big-omega of n*)

- An example

  $T(n) = 2n + 3$

  $2n \le T(n)$ for $n \ge 1$

  $cn \le T(n)$ for $n \ge 1$ and $c = 2$

  $T(n)$ belongs to $\Omega(n)$

$\Omega(n) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \le cn \le f(n)$ for all $n \ge n_0 \}$

Exercises

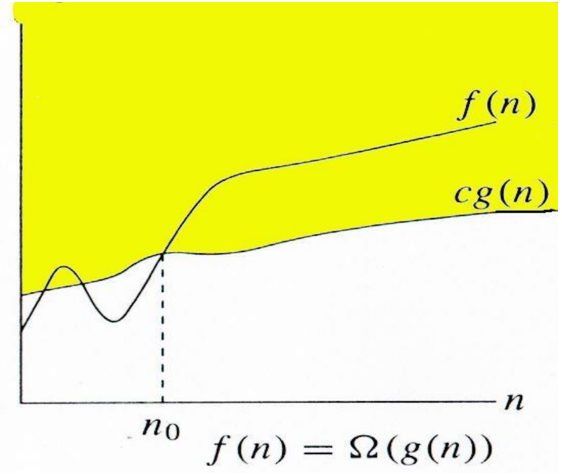
1. Is $2n + 1 \in \Omega(n)$?
2. Is $2n^2 + 10 \in \Omega(n^2)$?
3. Is $n^3 \in \Omega(n^2)$?

The set $\Omega(g(n))$

$\Omega(g(n)) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \leq c\, g(n) \leq f(n)$ for all $n \geq n_0 \}$

Ω-notation gives **a lower bound** for a function to within a constant factor

*$f(n) = \Omega(g(n))$, if there are positive constants **c and n_0** such that to the right of **n_0**, the value of **f(n) always lies on or above cg(n)**.*



Source: http://www.cs.unc.edu/~plaisted/comp122/02-asymp.ppt

The set $\Theta(n)$

- An example - $T(n) = 2n + 3$

$T(n) \le 6n$ for $n \ge 1$ $T(n)$ is $O(n)$

$2n \le T(n)$ for $n \ge 1$ $T(n)$ is $\Omega(n)$

$T(n)$ belongs to $\Theta(n)$

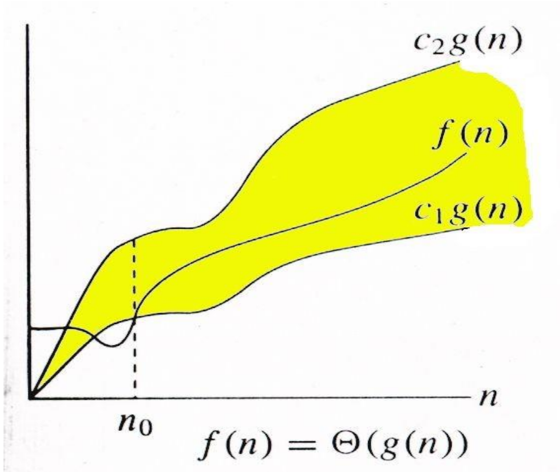
$\Theta(n) = \{ f(n)$: there exists positive constants c_1, c_2 and n_0 such that $0 \le c_1 n \le f(n) \le c_2 n$ for all $n \ge n_0 \}$

The set $\Theta(g(n))$

$\Theta(g(n)) = \{ f(n)$: there exists positive constants c_1, c_2 and n_0 such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0 \}$

Θ-notation gives **tight bound** for a function to within constant factors

$f(n)=\Theta(g(n))$, if there exists positive constants $\boldsymbol{c_1}$, $\boldsymbol{c_2}$ ***and*** $\boldsymbol{n_0}$ *such that to the right of* $\boldsymbol{n_0}$, *the value of $f(n)$ always lies between $c_1\, g(n)$ and $c_2\, g(n)$ inclusive.*



Source: http://www.cs.unc.edu/~plaisted/comp122/02-asymp.ppt

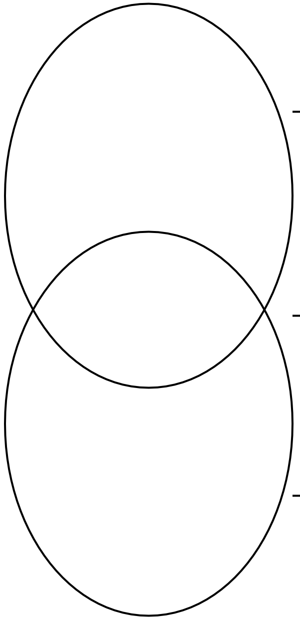
Asymptotic notations – Formal definitions

- $O(g(n)) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \leq f(n) \leq c\, g(n)$ for all $n \geq n_0 \}$

- $\Omega(g(n)) = \{ f(n)$: there exists positive constants c and n_0 such that $0 \leq c\, g(n) \leq f(n)$ all $n \geq n_0 \}$

- $\Theta(g(n)) = \{ f(n)$: there exists positive constants c_1, c_2 and n_0 such that $0 \leq c_1 g(n) \leq f(n) \leq c_2\, g(n)$ for all $n \geq n_0 \}$

To make it more clear



$\Omega(g)$: *functions that grow at least as fast as g*

$\Theta(g)$: *functions that grow at the same rate as g*

$O(g)$: *functions that grow no faster than g*

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

  $g(n)$ is an asymptotically tight bound for $f(n)$

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

  $g(n)$ is an asymptotic upper bound for $f(n)$

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

  $g(n)$ is an asymptotic lower bound for $f(n)$

Theorem

For any two functions $f(n)$ and $g(n)$, we have $f(n)=\Theta(g(n))$ if and only if $f(n)=O(g(n))$ and $f(n)=\Omega(g(n))$.

$$\boxed{\boldsymbol{\Theta(f(n)) = O(f(n)) \cap \Omega(f(n)).}}$$

Examples

- $f(n) = an^2+bn+c$, where a, b, and c are constants and $a>0$
- $f(n) = \Theta(n^2) \rightarrow f(n) = \Omega(n^2)$ *and* $f(n) = O(n^2)$

- **For any polynomial, $p(n)$ of degree k we have $p(n) = \Theta(n^k)$**
- **Any constant function is $\Theta(n^0)$, or $\Theta(1)$.**

Insertion Sort – Running Time

- Best Case running Time is $\Omega(n)$.

  Implies Running time on any input is $\Omega(n)$.
- Running time is not $\Omega(n^2)$.
- Worst Case running time is $\Omega(n^2)$.
- Is it correct to say best case running Time is $\Theta(n)$?

Is $O(n \lg n)$ algorithm preferred over $O(n^2)$?

- Suppose $T_1(n) \leq 50\, n \lg n$ and $T_2(n) \leq 2n^2$
- Check the values of $T_1(n)$ and $T_2(n)$ when n= 2 and n=1024
- For small input sizes, the $O(n^2)$ algorithm may run faster.
- Once the input size becomes large enough, $O(n \lg n)$ runs faster

  irrespective of the constant factors

  irrespective of the implementation.

- Read the corresponding Sections in CLRS.

References

1. (1) Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein*Introduction to Algorithms*, PHI, 2001.

2. (2) Sara Baase and Allen Van Gelder *Computer Algorithms: Introduction to Design & Analysis*, Pearson Education, third edition, 2000.

3. (3) Donald E Knuth. Big omicron and big omega and big theta. *ACM SIGACT*
4. *News*, 1976.

5. (4) Gilles Brassard, Paul Bratley, *Fundamentals of Algorithmics*, PHI, 1997.