BITS Pilani
Hyderabad Campus

Data Structures and Algorithms Design (DSECLZG519)

Febin.A.Vahab
Asst.Professor(Offcampus)
BITS Pilani,Bangalore

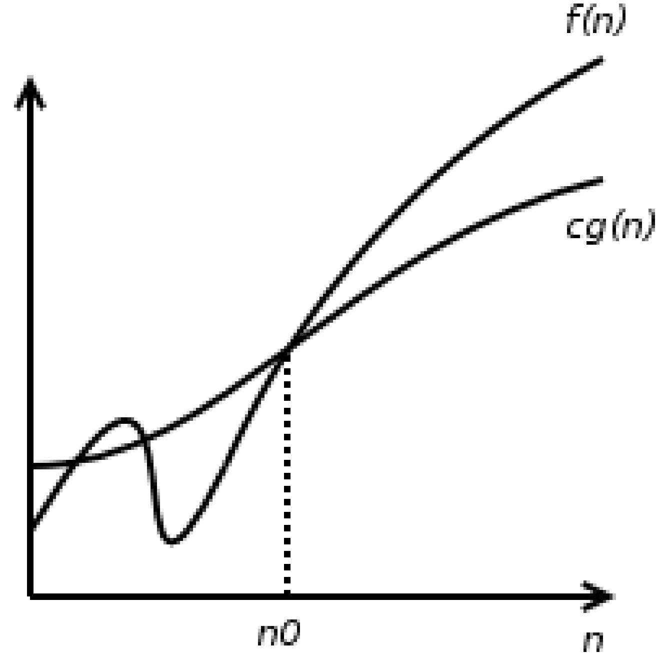
SESSION 3 -PLAN

Online Sessions(#)	List of Topic Title	Text/Ref Book/external resource
3	Big-Omega and Theta(Quick Review), Correctness of Algorithms.	T1: 1.2

Big-Omega Notation

- The function f(n) is said to be in Ω(g(n)) iff there exists a positive constant c and a positive integer n0 such that

f(n)>= c.g(n) for all n>=n0.



- Asymptotic lower bound
- $n^3 \in \Omega(n^2)$
- $n^5+n+3 \in \Omega(n^4)$

Big-Omega Notation

- Big-Omega notation provides a lower bound on a function to within a constant factor.
- To prove big-Omega, find witnesses, specific values for C and n0, and prove $n > n0$ implies $f(n) \geq C * g(n)$.

Tricks for Proving Big-Omega

- Assume $n > 1$ if you chose $n0 = 1$ (or $n > 10$ if you chose $n0 = 10$).
- To prove $f(n) \geq C * g(n)$, you need to find expressions smaller than $f(n)$ and larger than $C * g(n)$.
- If the lowest-order term is positive, just eliminate it to obtain a larger expression.
- Repeatedly use $-n0 > -n$ and $-0.1n0 > -0.1n$ and so on to "convert" the lowest-order term into a higher-order term.
- Check that your expressions are greater than $C * g(n)$ by using $n = 100$.

Tricks for Proving Big-Omega

- Generate a table for f(n) and g(n). using n = 1, n = 10 and n = 100.[Use values smaller than 10 and 100 if you wish.]
- Guess 1/C = [g(1)/f(1)] (or more likely 1/C = [g(10)/f(10)]).
- Check that $f(10) \geq C * g(10)$ and $f(100) \geq C * g(100)$.[If this is not true, f(n) might not be (g(n)).]
- Choose n0 = 1 (or n0 = 10).
- Prove that $\forall n(n > n0 \rightarrow f(n) \geq C * g(n))$.[It’s ok if you end up with a smaller, but still positive, value for C.]

Big-Omega Example 1

- Show that $n^2 - 2n + 1$ is $\Omega(n^2)$.
- In this case, $f(n) = n^2 - 2n + 1$ and $g(n) = n^2$.

n	f(n)	g(n)	Ceil(g(n)/f(n))	C
1	0	1	-	-
10	81	100	2	1/2
100	9801	10000	2	1/2

- This table suggests trying n0 = 10 and C = 1/2.

Big-Omega Example 1

- Try n0 = 10 and C = 1/2.
  - Want to prove n > 10 implies $n^2 - 2n + 1 \geq n^2/2$.
  - Assume n > 10. Want to show $f(n) \geq n^2/2$.
  - The lowest-order term is positive, so eliminate.
  - $n^2 - 2n + 1 > n^2 - 2n$
  - $n > 10$ implies $-10 > -n$, implies $-2 > -0.2n$.
  - $-2n > -0.2n^2$ implies $n^2 - 2n > n^2 - 0.2n^2 = 0.8n^2$.
  - $n > 10$ implies $0.8n^2 > n^2/2$.
  - This finishes the proof.

Big-Omega
More examples

- Show that $3n + 7$ is $\Omega(n)$.
- Show that $n^3/8 - n^2/12 - n/6 - 1$ is $O(n^3)$.

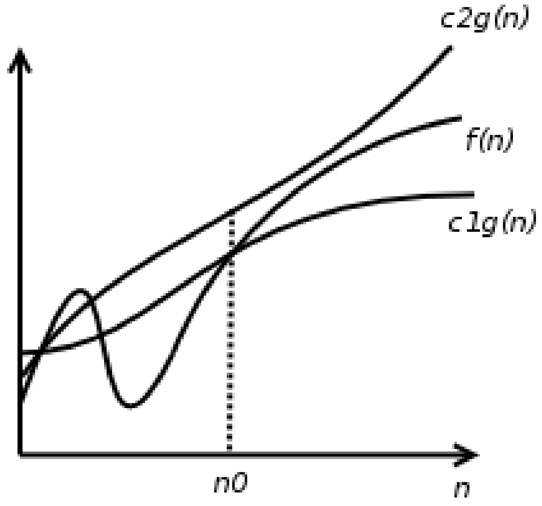
- Discuss in Canvas and check the solutions
- Solution present in slides(https://bits-pilani.instructure.com/groups/4548/pages/lecture-2-session-on-02-slash-05-slash-2020)

Big-Theta Notation

- The function f(n) is said to be in $\Theta(g(n))$ iff there exists some positive constants c1 and c2 and a non negative integer n0 such that

  c1.g(n)<=f(n)<=c2.g(n) for all n>=n0

- **Asymptotic tight bound**
- an^2+bn+c $\in \Theta$(n^2)
- n^2 $\in \Theta$(n^2)



Examples Θ

- **$f(n)=5n^2$ .Prove that $f(n)$ is $\Theta(n)$**
  - $5n^2=c.n$
  - $c.n=5n^2$
  - $c=5n$
  - If $n=1, c=5$
  - $5*1<=5*1$ hence the proof.

Little-Oh and little omega Notation

- f(n) is o(g(n)) (or f(n) ∈ o(g(n))) if for any real constant $c > 0$, there exists an integer constant $n0 \geq 1$ such that
  - ***f(n)* < *c* ∗ *g(n)*** for every integer $n \geq n0$.

- f(n) is ω(g(n)) (or f(n) ∈ ω(g(n))) if for any real constant $c > 0$, there exists an integer constant $n0 \geq 1$ such that
  - ***f(n)* > *c* * *g(n)*** for every integer $n \geq n0$.

Little-Oh and Little omega Notation

- $12n^2 + 6n$ is $o(n^3)$
- $4n+6$ is $o(n^2)$
- $4n+6$ is $\omega(1)$
- $2n^9 + 1$ is $o(n^{10})$
- n^2 is $\omega(\log n)$

USING LIMITS

Little Oh- $= \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

Little Omega $= \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

Correctness of algorithm

- An algorithm is said to be correct if, for every input instance, it halts with the correct output.
- When it can be incorrect?
  - Might not halt on all input instances
  - Might halt with an incorrect answer
- Does it makes sense to think of incorrect algorithm?
  - Might be useful if we can control the error rate and can be implemented very fast

THANK YOU!

BITS Pilani
Hyderabad Campus