Discrete Mathematics: Lectures 2 and 3
Asymptotic Notations

Instructor: Arijit Bishnu Date: July 22 and 23, 2009

1 Introductory Story

Asymptotic notations are mostly used in computer science to describe the asymptotic running time of an algorithm. As an example, an algorithm that takes an array of size n as input and runs for time proportional to n^2 is said to take $O(n^2)$ time. 'O' is pronounced as *big-oh*, so we say that the algorithm takes *big-oh of* n^2 time. Asymptotic notations also have their use in comparing the growth of functions.

The asymptotic notations, as we will see shortly, deal with functions that have $\mathbb{N}$ as their domain and $\mathbb{R}$, or mostly $\mathbb{R}_{\geq 0}$ as the range. The domain is $\mathbb{N}$ as the input size is a positive integer. But, after we go through the definitions, we can very well see that it will be applicable for functions where the domain is $\mathbb{R}$. This has to be clear from the context in which we are using asymptotic notations.

2 O (big-oh) notation: bounding from above

The O-notation is used for asymptotically upper bounding a function. Notice that there can be many functions that bound a particular function from above. We would use O (big-oh) notation to represent a set of functions that upper bounds a particular function.

Definition 1 *We say that a function $f(n)$ is big-oh of $g(n)$ written as $f(n) = O(g(n))$ if there exists positive constants c and n_0 such that $0 \leq f(n) \leq cg(n)$, $\forall n \geq n_0$. In terms of sets, $O(g(n))$ denotes a set of functions $f(n)$ that satisfies the above. Formally, $O(g(n)) =$*

$$\{f(n) \mid \exists \text{ positive constants } c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n), \forall n \geq n_0\}$$

A consequence of this definition in terms of limits is as follows. If $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists, then $\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq \infty$ implies $f(n) = O(g(n))$.

Example 1 *Let $f(n) = n^2$. Then, $f(n) = O(n^2)$, $f(n) = O(n^2 \log n)$, $f(n) = O(n^{2.5})$, $f(n) = O(n^3)$, $f(n) = O(n^4)$, ...*

Example 2 *Let $f(n) = 5.5n^2 - 7n$. We need to verify whether $f(n)$ is $O(n^2)$. Let c be a constant such that $5.5n^2 - 7n \leq cn^2$, or, $n \geq \frac{7}{c-5.5}$. Fix $c = 9$, to get $n \geq 2$. So, our $n_0 = 2$ and $c = 9$. This shows that there exists positive constants $c = 9$ and $n_0 = 2$ such that $0 \leq f(n) \leq cn^2$, $\forall n \geq n_0$.*

Exercise 1 *Let $f(n) = 5.5n^2 - 7n$. Verify whether $f(n) = O(n)$?*

Exercise 2 *Let $f(n) = a_k n^k + a_{k-1}n^{k+1} + \ldots + a_1 n^1 + a_o$ such that $a_k > 0$. Show that $f(n) = O(n^k)$.*

3 Ω (Omega) notation: bounding from below

The Ω-notation is used for asymptotically lower bounding a function. Notice that there can be many functions that bound a particular function from below. We would use Ω (big-omega) notation to represent a set of functions that lower bounds a particular function.

Definition 2 *We say that a function $f(n)$ is big-omega of $g(n)$ written as $f(n) = \Omega(g(n))$ if there exists positive constants c and n_0 such that $0 \leq cg(n) \leq f(n)$, $\forall n \geq n_0$. In terms of sets, $O(g(n))$ denotes a set of functions $f(n)$ that satisfies the above. Formally, $\Omega(g(n))$*

$$\{f(n) \mid \exists \text{ positive constants } c \text{ and } n_0 \text{ such that } 0 \leq cg(n) \leq f(n), \forall n \geq n_0\}$$

A consequence of this definition in terms of limits is as follows. If $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists, then $\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq 0$ implies $f(n) = \Omega(g(n))$.

Example 3 *Let $f(n) = 5.5n^2 - 7n$. We need to verify whether $f(n)$ is $\Omega(n^2)$. Let c be a constant such that $5.5n^2 - 7n \geq cn^2$, or, $n \geq \frac{7}{5.5-c}$. Fix $c = 3$, to get $n \geq 2.8$. So, our $n_0 = 2.8$ and $c = 3$. This shows that there exists positive constants $c = 3$ and $n_0 = 2.8$ such that $0 \leq cn^2 \leq f(n)$, $\forall n \geq n_0$.*

Exercise 3 *Let $f(n) = 5.5n^2 - 7n$. Verify whether $f(n) = \Omega(n^2)$. Verify whether $f(n) = \Omega(n)$.*

Exercise 4 *Let $f(n) = a_k n^k + a_{k-1}n^{k+1} + \ldots + a_1 n^1 + a_o$ such that $a_k > 0$. Show that $f(n) = \Omega(n^k)$. Verify whether $f(n) = \Omega(n^{k-1})$.*

Exercise 5 *Consider the following statement. $f(n)$ is $\Omega(g(n))$ if and only if $g(n)$ is $O(f(n))$. If you think the statement to be correct, prove it; else, disprove it.*

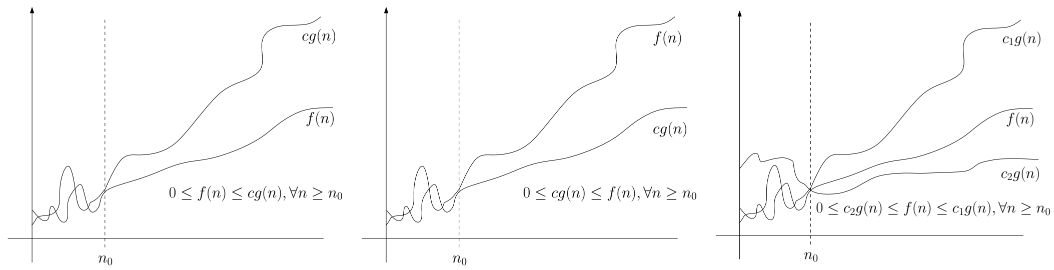
4 Θ (Theta) notation: bounding from above and below

The Θ-notation is used for asymptotically bounding a function from both above and below. Notice that there can be many functions that bound a particular function both from above and below. We would use Θ (theta) notation to represent a set of functions that bounds a particular function from above and below.

Definition 3 *We say that a function $f(n)$ is theta of $g(n)$ written as $f(n) = \Theta(g(n))$ if there exists positive constants c_1, c_2 and n_0 such that $0 \leq c_2 g(n) \leq f(n) \leq c_1 g(n)$, $\forall n \geq n_0$. In terms of sets, $O(g(n))$ denotes a set of functions $f(n)$ that satisfies the above. Formally, $\Theta(g(n)) =$*

$$\{f(n) \mid \exists \textit{ positive constants } c_1, c_2 \textit{ and } n_0 \textit{ such that } 0 \leq c_2 g(n) \leq f(n) \leq c_1 g(n), \forall n \geq n_0\}$$

A consequence of this definition in terms of limits is as follows. If $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists, then $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c$ implies $f(n) = \Theta(g(n))$ where c is a non-zero positive constant.



(a) $f(n) = O(g(n))$. (b) $f(n) = \Omega(g(n))$. (c) $f(n) = \Theta(g(n))$.

Figure 1: A diagramatic representation of the asymptotic notations O, Ω and Θ.

Exercise 6 *Let $f(n) = 5.5n^2 - 7n$. Verify whether $f(n) = \Theta(n)$?*

Example 4 *Any constant function is $O(1)$, $\Omega(1)$ and $\Theta(1)$. Can you prove it?*

Exercise 7 *For any two functions $f(n)$ and $g(n)$, show that $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.*

Solution: Hints: We want *if and only if.* To prove the above, you have to show that (i) $f(n) = \Theta(g(n))$ implies both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$ and (ii) $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$ implies $f(n) = \Theta(g(n))$. ■

Example 5 *Let $f(n) = 5.5n^2 - 7n$. We need to verify whether $f(n)$ is $\Theta(n^2)$. From Example 2 we have constants $c_1 = 9$ and $n_0 = 2$, such that $0 \leq f(n) \leq c_1 n^2$, $\forall n \geq n_0$. Similarly, we have from Example 3, we have constants $c_2 = 3$ and $n_0 = 2.8$, such that $0 \leq c_2 n^2 \leq f(n)$, $\forall n \geq n_0$. To show $f(n)$ is $\Theta(n^2)$, we have got hold of two constants c_1 and c_2. We fix the n_0 for Θ as maximum $\{2, 2.8\} = 2.8$. Thus, we have got positive constants c_1, c_2 and n_0 such that $0 \leq c_2 g(n) \leq f(n) \leq c_1 g(n)$, $\forall n \geq n_0$.*

The notations O, Ω and Θ have a special significance in the study of design and analysis of algorithms. We have already studied the problem of sorting and designed an $O(n \log n)$ algorithm by divide-and-conquer. It can also be shown [4] that the problem of sorting where only comparisons are used to determine the relative order of two numbers can be solved no faster than $cn \log n$, where c is a positive constant. We say that the problem of sorting has a lower bound of $\Omega(n \log n)$. Any algorithm of sorting that takes $O(n \log n)$ comparisons is said to be an optimal algorithm as asymptotically no faster algorithm can be obtained. We say that an algorithm for sorting taking $O(n \log n)$ comparisons that matches the lower bound of sorting, i.e. $\Omega(n \log n)$ is a $\Theta(n \log n)$ algorithm.

Exercise 8 *Prove that the running time of an algorithm is $\Theta(f(n))$ if and only if its worst-case running time is $O(f(n))$ and its best-case running time is $\Omega(f(n))$.*

5 o (small-oh) notation: bounding strictly from above

The O-notation is used for asymptotically upper bounding a function, but this notation may not be strict. Let $f(n) = n^2$. Then, $f(n) = O(n^2)$ is asymptotically tight but $f(n) = O(n^2 \log n)$, or $f(n) = O(n^{2.5})$, or $f(n) = O(n^3)$ are not asymptotically tight. The o (pronounced small-oh or little-oh) notation is used to denote those functions that are asymptotically strictly greater. Notice that there can be many functions that bound a particular function strictly from above.

Definition 4 *We say that a function $f(n)$ is small-oh of $g(n)$ written as $f(n) = o(g(n))$ if for any positive non-zero constant c (note the change from O), there exists a positive non-zero constant n_0 such that $0 \leq f(n) < cg(n)$, $\forall n \geq n_0$. In terms of sets, $o(g(n))$ denotes a set of functions $f(n)$ that satisfies the above. Formally, $o(g(n)) =$*

$$\{f(n) \mid \exists \textit{ constants } c > 0 \textit{ and } n_0 > 0 \textit{ such that } 0 \leq f(n) < cg(n), \forall n \geq n_0\}$$

A consequence of this definition in terms of limits is as follows. If $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists, then $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ implies $f(n) = o(g(n))$.

Example 6 *Let $f(n) = n^2$. Then, $f(n) = o(n^2 \log n)$, $f(n) = o(n^{2.5})$, $f(n) = o(n^3)$, $f(n) = o(n^4)$, $\ldots$, but $f(n) \neq o(n^2)$.*

Example 7 *Let $f(n) = 5.5n^2 - 7n$. Verify whether $f(n)$ is $o(n^2)$. Verify whether $f(n)$ is $o(n^2 \log n)$.*

6 ω (small-omega) notation: bounding strictly from below

The Ω-notation is used for asymptotically lower bounding a function, but this notation may not be strict. Let $f(n) = n^2$. Then, $f(n) = \Omega(n^2)$ is asymptotically tight but $f(n) = \Omega(n \log n)$, or $f(n) = \Omega(n)$, or $f(n) = \Omega(n^{\frac{1}{2}})$ are not asymptotically tight. The ω (pronounced small-omega or little-omega) notation is used to denote those functions that are asymptotically strictly smaller. Notice that there can be many functions that bound a particular function strictly from below. ω notation is to Ω notation as o notation is to O notation.

Definition 5 *We say that a function $f(n)$ is small-omega of $g(n)$ written as $f(n) = \omega(g(n))$ if for any positive non-zero constant c (note the change from Ω), there exists a positive non-zero constant n_0 such that $0 \leq cg(n) < f(n)$, $\forall n \geq n_0$. In terms of sets, $\omega(g(n))$ denotes a set of functions $f(n)$ that satisfies the above. Formally, $\omega(g(n)) =$*

$$\{f(n) \mid \exists \text{ constants } c > 0 \text{ and } n_0 > 0 \text{ such that } 0 \leq cg(n) < f(n), \forall n \geq n_0\}$$

A consequence of this definition in terms of limits is as follows. If $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists, then $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ implies $f(n) = \omega(g(n))$.

Example 8 *Let $f(n) = n^2$. Then, $f(n) = \omega(n)$, $f(n) = \omega(n \log n)$, $f(n) = \omega(n^{\frac{1}{3}})$, $\ldots$, but $f(n) \neq \omega(n^2)$.*

Example 9 *Let $f(n) = 5.5n^2 - 7n$. Verify whether $f(n)$ is $\omega(n^2)$. Verify whether $f(n)$ is $\omega(n \log n)$.*

Exercise 9 *Show that a function $f(n) \in \omega(g(n))$ if and only if $g(n) \in o(f(n))$.*

Exercise 10 *Show that $2^{n+1} = O(2^n)$.*

Exercise 11 *Verify whether $2^{2n} = O(2^n)$.*

Exercise 12 *Prove that $o(f(n)) \cap \omega(f(n)) = \emptyset$.*

Exercise 13 *Show that $n^{2+\epsilon} = o(2^n)$ and $n^{2-\epsilon} \neq o(2^n)$ where $\epsilon < 1$ is a small positive constant.*

Exercise 14 *$f(n) \prec g(n)$ denotes $f(n) = o(g(n))$. Using this notation, find the hierarchy of the following functions: $\log^2 n$, 2^{n^2}, $\log \log n$, $n!$, 2^n, $n^{4/5}$, $\sqrt{n}$; and fill up the following table.*

	$\prec$		$\prec$		$\prec$		$\prec$		$\prec$		$\prec$	

Exercise 15 *Let $f_1(n)$ and $f_2(n)$ be two non-negative functions in n, where n is a positive integer. Suppose $f_1(n) = O(f_2(n))$.*

(i) Show that $f_2(n) = \Omega(f_1(n))$.

(ii) Consider the statement: $2^{f_1(n)} = O(2^{f_2(n)})$. *If it is true, prove it; else, disprove it.*

Exercise 16 *Let* $f(n)$ *and* $g(n)$ *be asymptotically positive functions. Prove the following:*

Transitivity: $f(n) = \mathcal{X}(g(n))$ *and* $g(n) = \mathcal{X}(h(n))$ *imply* $f(n) = \mathcal{X}(h(n))$ *where* $\mathcal{X} = \{O, \Omega, \Theta, o, \omega\}$.

Reflexivity: $f(n) = \mathcal{X}(f(n))$ *where* $\mathcal{X} = \{O, \Omega, \Theta\}$.

Symmetry: $f(n) = \Theta(g(n))$ *if and only if* $g(n) = \Theta(f(n))$.

Transpose Symmetry: $f(n) = O(g(n))$ *if and only if* $g(n) = \Omega(f(n))$.

Transpose Symmetry: $f(n) = o(g(n))$ *if and only if* $g(n) = \omega(f(n))$.

Exercise 17 *The assymptotic notation* O *satisfies the transitive property, i.e. if* $f(n) = O(g(n))$ *and* $g(n) = O(h(n))$, *then* $f(n) = O(h(n))$. *Now, if* $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ *exists, then* $\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq \infty$ *implies* $f(n) = O(g(n))$. *Now, let* $f(n) = 2^{n+1}$ *and* $g(n) = 2^n$. *Then,* $\lim_{n\to\infty} \frac{2^{n+1}}{2^n} = 2 \neq \infty$. *So,* $2^{n+1} = O(2^n)$. *Extending this further, we can write* $2^n = O(2^{n-1})$, $\ldots$, $2^i = O(2^{i-1})$, $\ldots$ *So, using the transitive property, we can write* $2^{n+1} = O(2^{i-1})$. *We can go on extending this, so that finally* $2^{n+1} = O(2^k)$, *where* k *is a constant. So, we can write* $2^{n+1} = O(1)$. *Do you agree to what has been proved? If not, where is the fallacy?*

References

[1] J. Matoušek and J. Nešetřil, *Invitation to Discrete Mathematics*, Oxford University Press, New York, 1998.

[2] C. L. Liu, *Elements of Discrete Mathematics*, Tata McGraw Hill, New Delhi, 2000.

[3] Ronald L. Graham, Donald E. Knuth and O. Patashnik, *Concrete Mathematics*, Pearson Education,

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein , *Algorithms - Design Techniques and Analysis*, Prentice Hall of India.