ICS 353–Handout 2

Asymptotic Notations

Landau symbols

Very useful for comparing the performances of algorithms with respect to the consumed time and space. That is, they are used to describe the complexity classes of algorithms.

Definitions

Let $f(n),\ g(n) : \mathbb{N} \to (0, \infty)$ be two functions. Then

1. **Big-Oh:** $f(n) = O(g(n)) \iff \exists\ n_1 \in \mathbb{N}$ and a constant $c_1 > 0$ such that

$$f(n) \le c_1\ g(n), \qquad \text{for all} \quad n \ge n_1\,.$$

2. **Big-Omega:** $f(n) = \Omega(g(n)) \iff \exists\ n_2 \in \mathbb{N}$ and a constant $c_2 > 0$ such that

$$f(n) \ge c_2 g(n), \qquad \text{for all} \quad n \ge n_2\,.$$

3. **Theta:** $f(n) = \Theta(g(n)) \iff \exists\ n_0 \in \mathbb{N}$ and two constants $c_1, c_2 > 0$ such that

$$c_1 g(n) \le f(n) \le c_2 g(n), \qquad \text{for all} \quad n \ge n_0\,.$$

4. **Small-oh:** $f(n) = o(g(n)) \iff$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0\,.$$

We write $f(n) \ll g(n)$ or $f(n) \prec g(n)$.

5. **Small-omega:** $f(n) = \omega(g(n)) \iff$

$$\lim_{n \to \infty} \frac{g(n)}{f(n)} = 0\,.$$

We write $f(n) \gg g(n)$ or $f(n) \succ g(n)$..

Remarks

Notice the following.

1. **Big Oh** = upper bound of a function.

   Used to get an upper bound on the worst-case (or the maximum) running time.

2. **Big Omega** = lower bound of a function.

   Used to get a lower bound on the best-case (or the minimum) running time of the algorithm.

3. $\Theta = O + \Omega$

   $f(n) = \Theta(g(n)) \iff f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

   This means both functions are of the same order; in fact

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{g(n)}{f(n)} = constant\,.$$

4. If $f(n) = o(g(n))$ or $g(n) = \omega(f(n))$, we write $f(n) \prec g(n)$. This means that the functions belong to different classes, indeed, $g(n)$ goes to ∞ faster than $f(n)$, as $n \to \infty$.

   Examples:

$$1 \prec \log^* n \prec \log\log n \prec \sqrt{\log n} \prec \frac{\log n}{\log\log n} \prec$$

$$\log n \prec \sqrt{n} \prec n \prec n\log n \prec n^2 \prec e^n \prec n! \prec n^n\,.$$

5. In all of these cases, we only need to prove these statements for n large enough (i.e., for $n \geq n_0$). That's why it is called asymptotic notations.

6. The constants c_1 and c_2 are all hidden within the notations because they are not important at this stage.

   Example: Suppose the running time of a given algorithm is $O(n^2)$. Say we run the algorithm with input of size $n = 100$, and we find that it takes 4 seconds. If we want to find the time for $m = 10n$, we don't have to run the algorithm over again. Since the time grows quadratically in this case, the estimated time should be close to 100×4 seconds. Had this algorithm been linear $O(n)$, the time would have been 10×4!

7. Thus, the performance analysis of algorithms should be independent from the type of machine and technology. I.e.,

   - We should concentrate on the asymptotic performances (for large input size n),
   - We should concentrate on the main term and ignore the smaller ones and the constant factors,
   - The constant factors and other smaller terms are useful only to compare between two algorithms that have the same order of running time.

Examples

1. Let $f(n) = 35n$ and $g(n) = 2n + 3$. Then $f(n) = \Theta(g(n))$ because

$$1 \times g(n) \leq f(n) \leq 20 \times g(n), \quad \forall \; n \geq 1\,.$$

2. For any constants $a > 0$ and b, we have $f(n) = an + b = \Theta(n)$. Notice also that $f(n) = O(n^2) = O(n^3) = O(n^k)$ for any $k > 1$ because $an + b \leq (a + b)n^k$, for any $k > 1$ and $n \geq 1$.

3. If $f(n) = 5n^2 - 6n + 3$ and $g(n) = 2n + 8$, then $f(n) = \omega(g(n))$ and $g(n) = o(f(n))$ because

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{2n+8}{5n^2-6n+3} = \lim_{n\to\infty} \frac{2+8/n}{5n-6+3/n} = 0\,.$$

4. Clearly, $\log n = O(n)$ and $n = \Omega(\log n)$. In fact, $\log n = o(n)$ and $n = \omega(\log n)$ because by L'Höpital's rule

$$\lim_{n\to\infty} \frac{\log n}{n} = 0\,.$$

In fact, if $c \in (0,1)$ is any constant than $\log n = o(n^c)$ by applying the same rule.

5. Also, $\log n^k = k \log n = o(n)$, for any constant $k > 0$, and $n + \log n^k = \Theta(n)$ because $n \le n + k \log n \le 2kn$.

6. $n + \sqrt{n}\log n = \Theta(n)$ because $\log n \le \sqrt{n}$ and hence $n \le n + \sqrt{n}\log n \le 2n$.

7. Clearly, for any constant $c \in (0, 1)$ we have

$$n = \omega(n^c) = \omega(\log n) = \omega(\log \log n) = \omega(1) .$$

8. Also, $c^n = O(n!) = O(n^n)$, for any constant $c > 0$, and $\log n! = \Theta(n \log n)$.

9. See also Examples 1.12- 1.14 in the textbook.