csce750 — Analysis of Algorithms
Fall 2020 — Lecture Notes: Asymptotic Notations

This document contains slides from the lecture, formatted to be suitable for printing or individual reading, and with some supplemental explanations added. It is intended as a supplement to, rather than a replacement for, the lectures themselves — you should not expect the notes to be self-contained or complete on their own.

1 Asymptotic notations

CLRS 3

In the analysis of algorithms, we are usually interested in how the performance of our algorithm changes *as the problem size increases*.

The primary tools for measuring the growth rate of a function that describes the run time of an algorithm are the **asymptotic notations**.

This provides a way of studying the algorithms themselves, independent of any specific hardware, operating system, compiler, programmer, *etc*.

2 Big-O

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

3 Big-Ω

$\Omega(g(n)) = \{f(n) \mid$ there exist positive constants c and n_0,
such that $0 \le cg(n) \le f(n)$ for all $n \ge n_0$. $\}$

4 Big-Θ

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

More succinctly: $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$

5 Anonymous functions

These notations officially refer to sets of functions, but it's often useful to use them in larger arithmetic expressions.
$T(n) = O(n^2)$ means:

$T(n) \in O(n^2)$

$T(n) = 2n^2 + O(n)$ means:

There exists $f(n) \in O(n)$ with $T(n) = 2n^2 + f(n)$.

6 *Little-o*

$o(g(n)) = \{f(n) \mid$ for any positive constant c,
there exists a constant n_0 such that $0 \leq f(n) \leq cg(n)$
for all $n \geq n_0$. $\}$

This indicates a loose bound: $f(n) = o(g(n))$ means $f(n)$ grows strictly slower than $g(n)$.

7 *Little-ω*

$\omega(g(n)) = \{f(n) \mid$ for any positive constant c,
there exists a constant n_0 such that $0 \leq cg(n) \leq f(n)$
for all $n \geq n_0$. $\}$

This indicates a loose bound: $f(n) = \omega(g(n))$ means $f(n)$ grows strictly faster than $g(n)$.

8 *Little-θ?*

9 *Limits*

For functions that are eventually positive, we can compare asymptotic growth rates using limits.

Let $L = \lim_{n\to\infty} \frac{f(n)}{g(n)}$, if that limit exists.

Then $f(n)$ is in ...

L	$\omega(g(n))$?	$\Omega(g(n))$?	$\Theta(g(n))$?	$O(g(n))$?	$o(g(n))$?
0				✗	✗
$(0, \infty)$		✗	✗	✗	
∞	✗	✗			

10 *Informal summary of intuition*

- $f(n) = O(g(n))$ is like $a \leq b$.
- $f(n) = \Omega(g(n))$ is like $a \geq b$.
- $f(n) = \Theta(g(n))$ is like $a = b$.
- $f(n) = o(g(n))$ is like $a < b$.
- $f(n) = \omega(g(n))$ is like $a > b$.