

Chapter 1: Algorithm Complexity Analysis

Anna Brandenberger, Anton Malakhveitchouk

January 26, 2018

This is the first chapter of the augmented transcript of a lecture given by Luc Devroye on the 11th of January 2018 for the Honours Data Structures and Algorithms class (COMP 252, McGill University). The subject was the definition of terms used in algorithm complexity analysis.

Landau Symbol Definitions

Landau symbols¹ are used to evaluate and to concisely describe the performance of an algorithm in time and space. In what follows, a_n and b_n are sequences of positive numbers.

Definition 1. O - Big Oh provides an upper bound.

We say that $a_n = O(b_n)$ if $\exists c > 0, n_0$ such that $a_n \leq cb_n, \forall n \geq n_0$.

Definition 2. Ω - Uppercase Omega describes a lower bound of a function. We say that $a_n = \Omega(b_n)$ if $\exists c > 0, n_0 > 0$ such that $a_n \geq cb_n, \forall n \geq n_0$.

Definition 3. Θ - Uppercase Theta is used to describe the precise asymptotic behaviour of a function. It combines both Big Oh and Uppercase Omega. We say that $a_n = \Theta(b_n)$ if $a_n = O(b_n) = \Omega(b_n)$.

Definition 4. o - Little oh

We write $a_n = o(b_n)$ when $\lim_{n \rightarrow \infty} \frac{a_n}{b_n} = 0$.

Definition 5. ω - Lowercase omega

We use $a_n = \omega(b_n)$ to denote $\lim_{n \rightarrow \infty} \frac{a_n}{b_n} = \infty$.

In the context of algorithms, Big Oh is used to give upper bounds on the complexity, and Big Omega is employed when the worst-case average time of an algorithm needs to be bounded from below. O gives "good news" and Ω is the carrier of "bad news".

Landau symbols hide the constant c from us.

¹ Cormen et al. [1989] Definitions 1 - 7.

Example 1: $a_n = 5n^4 + 3n^2 - 7n + 9$
 $5n^4 + 3n^2 - 7n + 9 \leq 17n^4$
Ignoring the constant, we see that the function is bounded by $O(n^4)$.

Example 2: $5n^4 + 3n^2 - 7n + 9$
This function is $\Omega(n^4)$.

Example 3: As an example, let $a_n = c_k n^k + c_{k-1} n^{k-1} + \dots + c_0$ be a polynomial of degrees $k > 0$ with $c_k > 0$. Then it is easy to show (do it!) that $a_n = \Theta(n^k)$, $a_n = o(n^{k+1})$ and $a_n = \omega(n^{k-1})$.

Complexity Classes

Complexity classes can be ordered from smallest to largest according to the worst-case times of various algorithms. A summary table follows:

In between these classes there exists a sea of possible algorithm complexities

1	Constant Time	
$\log_2 n$	Logarithmic	(<i>e.g., binary search</i>)
...		
n	Linear	
$n \log_2 n$	Linearithmic Time	(<i>e.g., mergesort</i>)
n^2	Quadratic	(<i>e.g., bubble sort</i>)
...		
2^n	Exponential	(<i>e.g., password guess</i>)
...		
$n!$	Factorial	(<i>e.g., traveling salesman problem</i>)
...		
2^{2^n}	Tower Functions	
...		

} Polynomial
 } Exponential

Sedgewick and Wayne [2005]

Let T_n denote the worst-case or average complexity of an algorithm on input of size n .

Definition 6. An algorithm is said to run in **polynomial time** if $\exists t, c > 0$ such that $T_n \leq cn^t$.

Definition 7. An algorithm is said to run in **exponential time** if $\exists \alpha, \alpha' > 0$ and $c, c' > 1$ such that $\alpha c^n \leq a_n \leq \alpha' c'^n$.

References

T.H Cormen, C.E. Leiserson, R.L.Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 1989. ISBN 9780262033848.

R. Sedgewick and K. Wayne. *Introduction to Programming in Java*. Pearson, 1st edition, 2005.