

ENG 346

Data Structures and Algorithms for Artificial Intelligence

Runtime Complexity of the Algorithms

Dr. Mehmet PEKMEZCİ

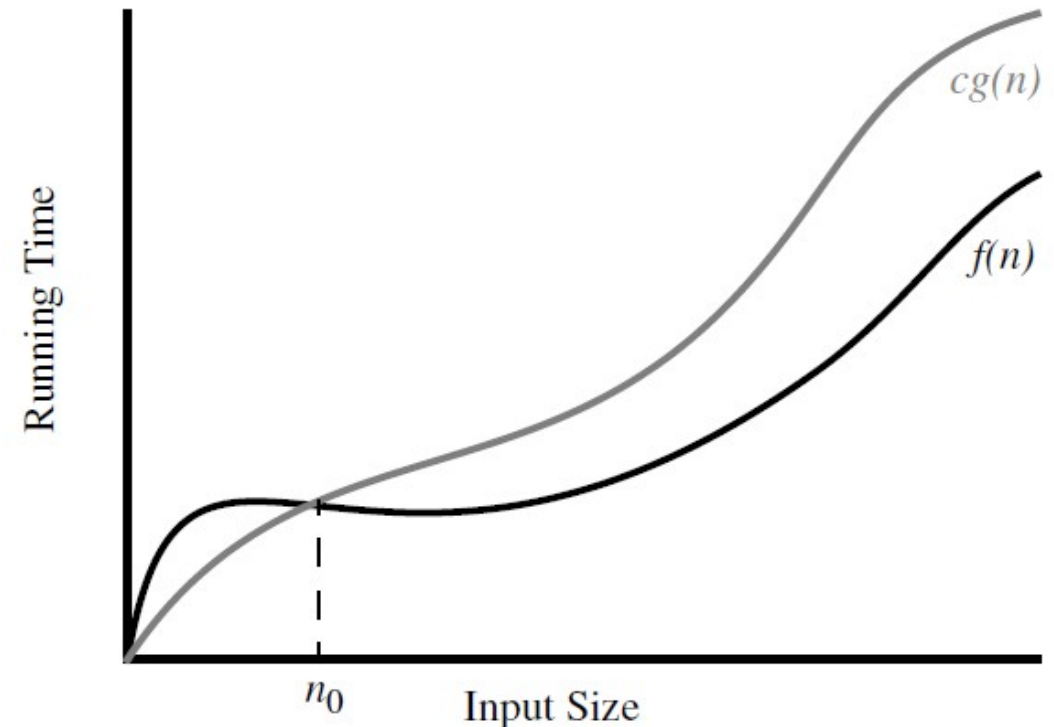
mpekmezci@gtu.edu.tr

<https://github.com/mehmetpekmezci/GTU-ENG-346>

Definitions: Big O

- Worst Case Scenario
- Upper-bound of a function $f(n)$
- Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers. We say that $f(n)$ is $O(g(n))$ if
 - there is a real constant $c > 0$ and
 - an integer constant $n_0 \geq 1$ such that

$$f(n) \leq c g(n), \text{ for } n \geq n_0.$$
- $f(n)$ is $O(g(n))$



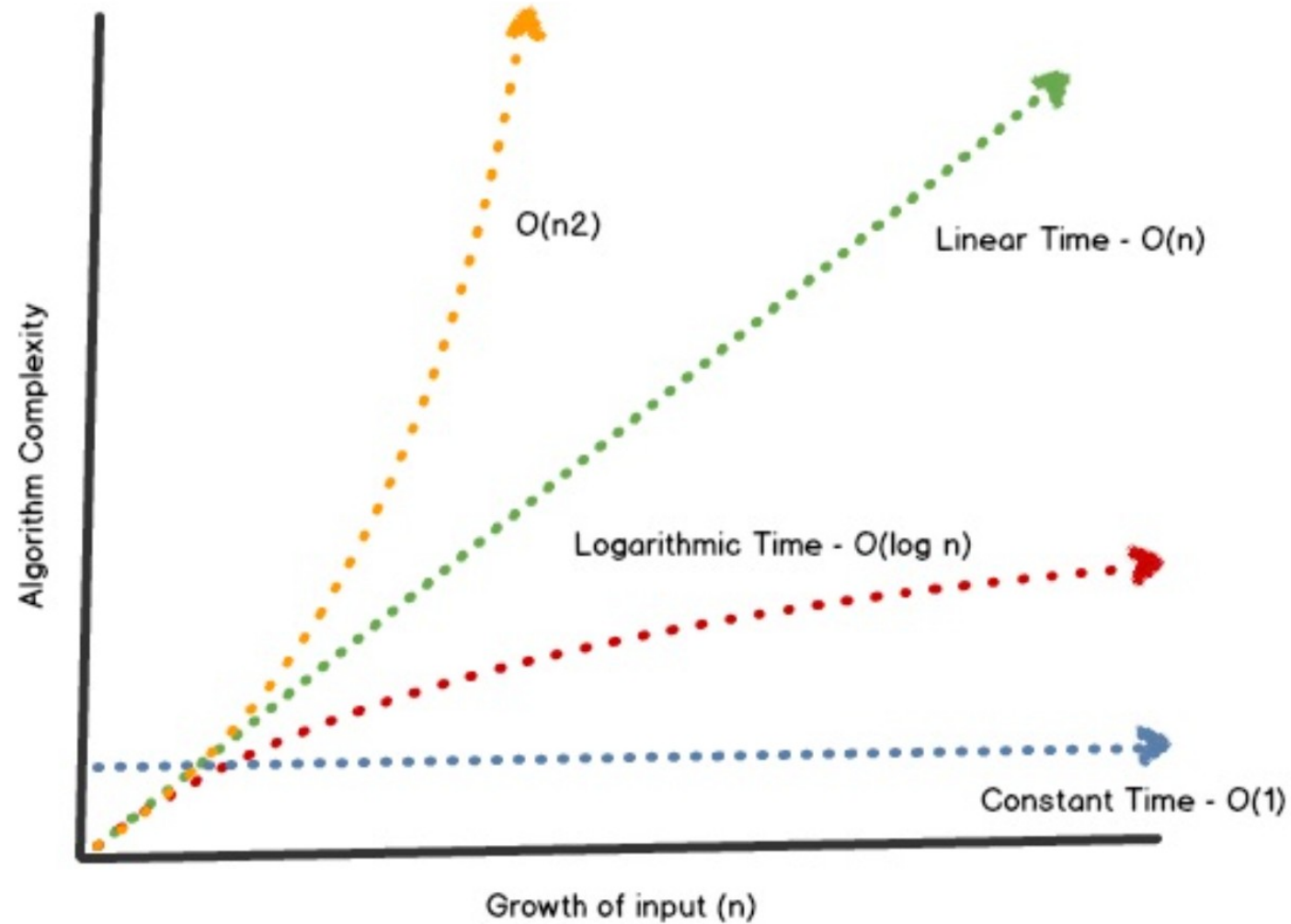
Big O Rules

- Simplifications:
- If $f(n)$ is a polynomial of degree d , then $f(n)$ is $O(n^d)$, i.e.,
 - Drop lower-order terms
 - Drop constant factors
- Use the smallest possible class of functions
 - Say “ $2n$ is $O(n)$ ” instead of “ $2n$ is $O(n^2)$ ”
- Use the simplest expression of the class
 - Say “ $3n + 5$ is $O(n)$ ” instead of “ $3n + 5$ is $O(3n)$ ”

Basics

Name	Function	Relation	Example
Constant Time	$f(n) = c$	Does not depend on input size.	Accessing array elements.
Logarithmic Time	$f(n) = \log n$	Running time increases logarithmically with the input size.	Binary search.
Linear Time	$f(n) = n$	Running time increases linearly with the input size.	Iterating through an array or list.
Linearithmic Time	$f(n) = n \log n$	The running time grows slower than $O(n^2)$ but faster than $O(n)$.	Efficient sorting algorithms like quicksort and mergesort.
Quadratic Time	$f(n) = n^2$	Running time grows proportionally to the square of the input size.	Algorithms with nested loops, such as selection sort or bubble sort.
Polynomial Time	$f(n) = n^k$	Running time is a polynomial function of the input size.	Algorithms with “k” nested loops.
Exponential Time	$f(n) = 2^n$	Running times that grow very rapidly with the input size.	N-P complete problems, such as traveling salesman.

Growth Rates



Examples:

- $7n-2$ is $O(n)$
- $3n^3 + 20n^2 + 5$ is $O(n^3)$
- $3 \log n + 5$ is $O(\log n)$

Exercises

- Book: R-3.1