Big O: Time Complexities for various data structures and algorithms

- The term complexity refers to either run time or space or both. If referenced without any qualifier, it means run time complexity
- Run time complexities is found using approximations based on the number of operation performed as the size of the input grows.
- Unless otherwise stated, run time complexities refer to the worst case run time.

Some typical run time orders are:

O(1): Constant	O(n): Linear	O(n^2): Quadratic	O(n^3): Cubic	O(log n): Logarithmic	O(2^n): Exponential	O(n!): Factorial

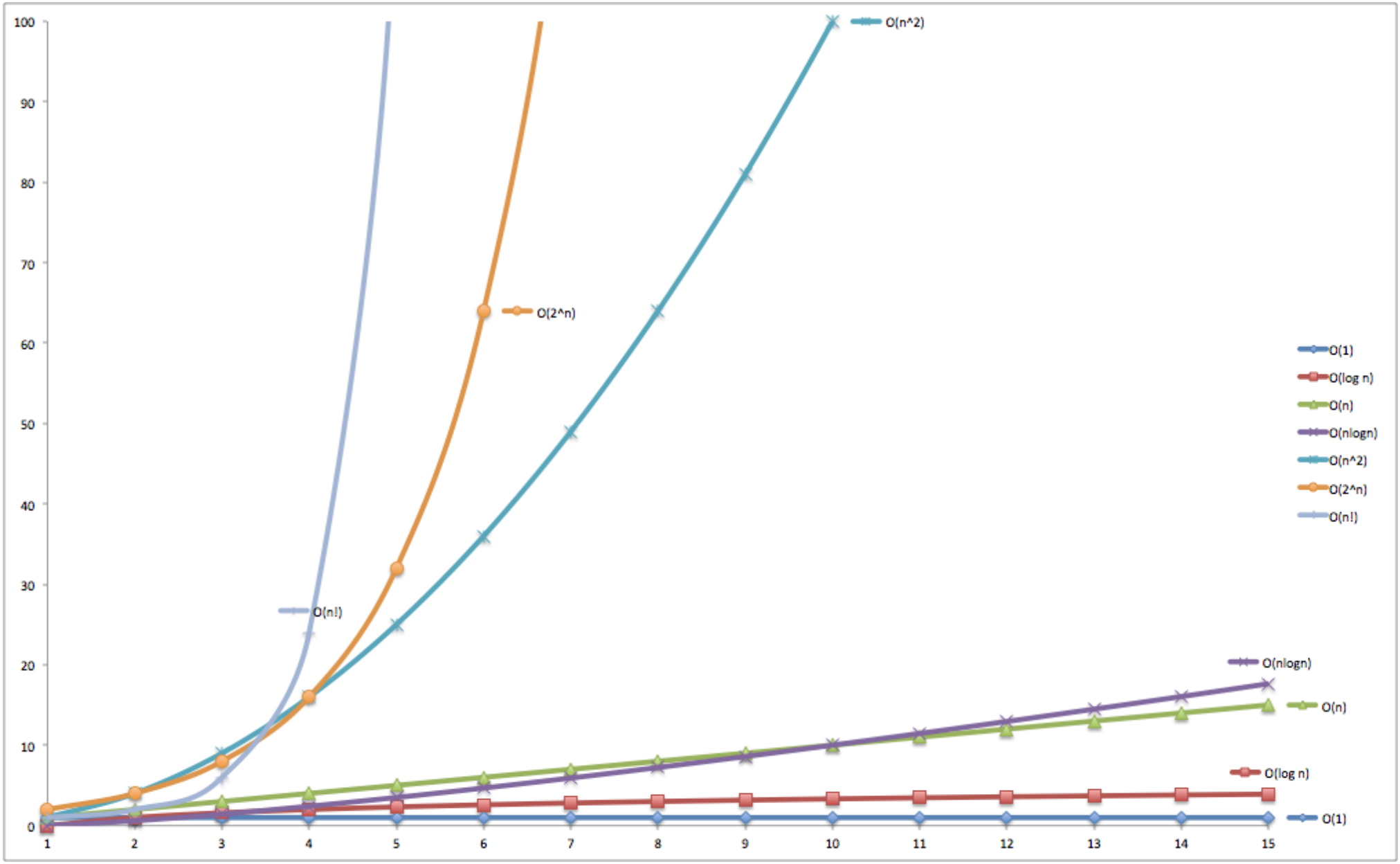
	Best Case				Average Case				Worst Case			
Data Structures	Access	Insert	Delete	Find	Access	Insert	Delete	Find	Access	Insert	Delete	Find
Arrays	O(1)	O(n) O(1)*	O(n) O(1)*	O(n)	O(1)	O(n) O(1)*	O(n) O(1)*	O(n)	O(1)	O(n) O(1)*	O(n) O(1)*	O(n)
Linked List	O(n)	O(1)** O(n)	O(1)** O(n)	O(n)	O(n)	O(1)** O(n)	O(1)** O(n)	O(n)	O(n)	O(1)** O(n)	O(1)** O(n)	O(n)
Stack	O(n)	O(1)	O(1)	***	O(n)	O(1)	O(1)	***	O(n)	O(1)	O(1)	***
Queue	O(n)	O(1)	O(1)	***	O(n)	O(1)	O(1)	***	O(n)	O(1)	O(1)	***
Hashmaps	O(1)	O(1)	O(1)	O(1)	O(1)	O(1)	O(1)	O(1)	O(n)	O(n)	O(n)	O(n)
Binary Search Trees	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(n)	O(n)	O(n)	O(n)

* If append or delete at the end of the array, it will be O(1), otherwise O(n) due to reassigning indices. Although O(1), due to fixed memory allocation, if resize needed, these operations may become O(n)
** If an unordered list then insertion and deletion is implemented as either at the head or tail making it O(1). If ordered list, O(n)
*** Not common operations in stack and queue. If a search functionality is needed, use a different data structure

Algorithms	Best Case	Average Case	Worst Case
Linear Search	O(1)	O(n)	O(n)
Binary Search	O(log n)	O(log n)	O(log n)
Depth First Search (DFS)	O(n)	O(n)	O(n)
Breadth First Search (BFS)	O(n)	O(n)	O(n)
Bubble Sort	O(n)	O(n^2)	O(n^2)
Merge Sort	O(n log n)	O(n log n)	O(n log n)
Quick Sort	O(n log n)	O(n log n)	O(n^2)

Big O

O(1): Constant	O(n): Linear	O(n^2): Quadratic	O(n^3): Cubic	O(log n): Logarithmic	O(2^n): Exponential	O(n!): Factorial



Number of input

pub: 2018-01-26