# Big O Notation

**By Jose Taveras**
**Whiteboard Wednesday Lecture 12/9/2015**

# PRESENTATION GOALS

1. Define Big-O notation and convince you why you should spend some time understanding it.
2. Discuss the terminology and get a bit mathy.
3. Practice whiteboarding!

**Big O?... oh…**
**I know...**



Close?… not really

# Algorithm Analysis

“*In computer science, the analysis of algorithms is the determination of the amount of resources (such as time and storage) necessary to execute them.*” - Wikipedia

‘In sorting n objects, merge sort has an average and worst-case performance of O(n log n).’

is pronounced

‘...performance of oh-of-en-log-en’

# Big O Notation

"*Big O Notation (with a capital letter O, not a zero), also called Landau's symbol, is a symbolism used in complexity theory, computer science, and mathematics to describe the asymptotic behavior of functions. Basically, it tells you how fast a function grows or declines.*" - Big O Notation Handout for MIT 16.070 Computer Science Course

'Why is the Big-O complexity of this algorithm O(n^2)?'

is pronounced

'Why is the big-oh complexity of this algorithm oh-of-en-squared?'

# FORMAL DEFINITIONS AND USAGE EXAMPLES

“The term asymptotic means approaching a value or curve arbitrarily closely. A line or curve A that is asymptotic to given curve C is called asymptote of C.”- Wolfram Mathworld Website

‘A is an asymptote of C’

‘My function is an asymptote of n^2’

‘My function is O(n^2)’

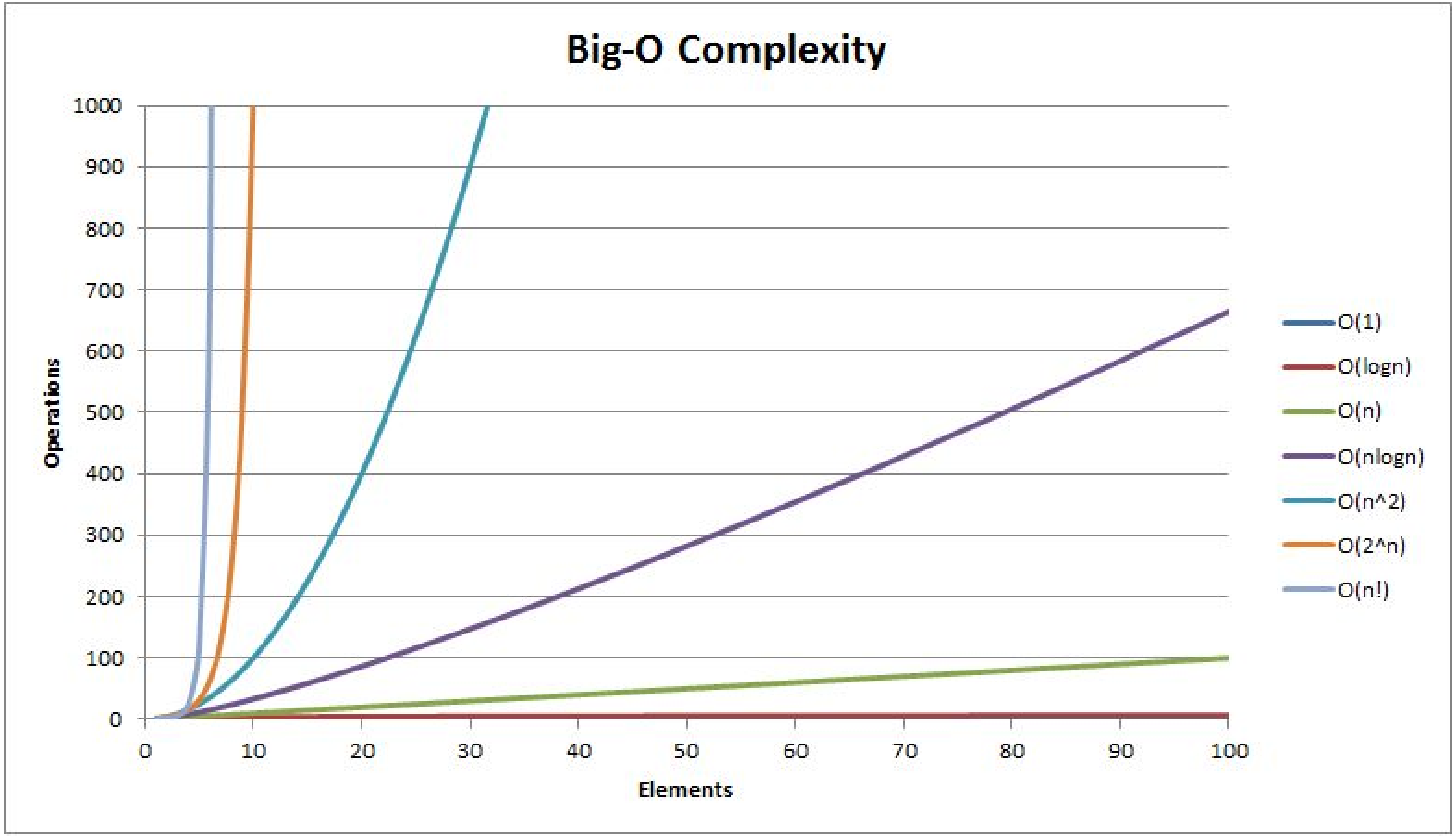
# Function classes

| notation | name |
|---|---|
| O(1) | constant |
| O(log(n)) | logarithmic |
| O((log(n)) c ) | polylogarithmic |
| O(n) | linear |
| O(n^2 ) | quadratic |
| O(n^c ) | polynomial |
| O(c^n ) | exponential |

# Data Structure Operations

| Data Structure | Time Complexity | | | | | | | | Space Complexity |
|---|---|---|---|---|---|---|---|---|---|
| | Average | | | | Worst | | | | Worst |
| | Access | Search | Insertion | Deletion | Access | Search | Insertion | Deletion | |
| Array | O(1) | O(n) | O(n) | O(n) | O(1) | O(n) | O(n) | O(n) | O(n) |
| Stack | O(n) | O(n) | O(1) | O(1) | O(n) | O(n) | O(1) | O(1) | O(n) |
| Singly-Linked List | O(n) | O(n) | O(1) | O(1) | O(n) | O(n) | O(1) | O(1) | O(n) |
| Doubly-Linked List | O(n) | O(n) | O(1) | O(1) | O(n) | O(n) | O(1) | O(1) | O(n) |
| Skip List | O(log(n)) | O(log(n)) | O(log(n)) | O(log(n)) | O(n) | O(n) | O(n) | O(n) | O(n log(n)) |
| Hash Table | - | O(1) | O(1) | O(1) | - | O(n) | O(n) | O(n) | O(n) |
| Binary Search Tree | O(log(n)) | O(log(n)) | O(log(n)) | O(log(n)) | O(n) | O(n) | O(n) | O(n) | O(n) |
| Cartesian Tree | - | O(log(n)) | O(log(n)) | O(log(n)) | - | O(n) | O(n) | O(n) | O(n) |
| B-Tree | 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) |
| Red-Black Tree | 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) |
| Splay Tree | - | O(log(n)) | O(log(n)) | O(log(n)) | - | O(log(n)) | O(log(n)) | O(log(n)) | O(n) |
| AVL Tree | 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) |

# FUNCTION CLASSES



# Further into algorithm analysis

| Notation | Definition | Analogy |
|---|---|---|
| f(n) = O(g(n)) | | <= or “at least as good as” |
| f(n) = o(g(n)) | | < or “definitely better than” |
| f(n) = Ω(g(n)) | g(n)=O(f(n)) | >= |
| f(n) = ω(g(n)) | g(n)=o(f(n)) | > |
| f(n) = θ(g(n)) | f(n)=O(g(n)) and g(n)=O(f (n)) | = |

# Example analysis: bubble sort

```
def bubble_sort(array)
  n = array.length
  loop do
    swapped = false

    (n-1).times do |i|
      if array[i] > array[i+1]
        array[i], array[i+1] = array[i+1], array[i]
        swapped = true
      end
    end

    break if not swapped
  end

  array
end
```

Considerations

- Time Complexity
- Space Complexity
- Best Case?
- Worst Case?

6 5 3 1 8 7 2 4

# References


- *The Big O*
  - http://www.sunrise-inc.co.jp/international/
- Algorithm Analysis, Retrieved from Wikipedia on 12/8/15
  - https://en.wikipedia.org/wiki/Analysis_of_algorithms
- Merge Sort, Retrieved from Wikipedia on 12/8/15
  - https://en.wikipedia.org/wiki/Merge_sort
- *MIT Computer Science Course: Introduction to Computers and Programming*
  - http://web.mit.edu/16.070/www/lecture/big_o.pdf
- Big O Discussion on Usage Example
  - http://stackoverflow.com/questions/33858889/why-is-the-big-o-complexity-of-this-algorithm-on2
- Asymptotic
  - http://mathworld.wolfram.com/Asymptotic.html
- Function Classes Plot Diagram
  - http://bigocheatsheet.com/
- Function Class Plot
  - https://www.desmos.com/calculator/orpnzoqequ
- Merge Sort Author's Blog
  - http://markmiyashita.com/interviews/problems/merge_sort/
- Bubble Sort Implementation
  - http://www.sitepoint.com/sorting-algorithms-ruby/


comments?
questions?

Bonus

6 5 3 1 8 7 2 4