Big O: A Review

By: Geir Anderson

Outline

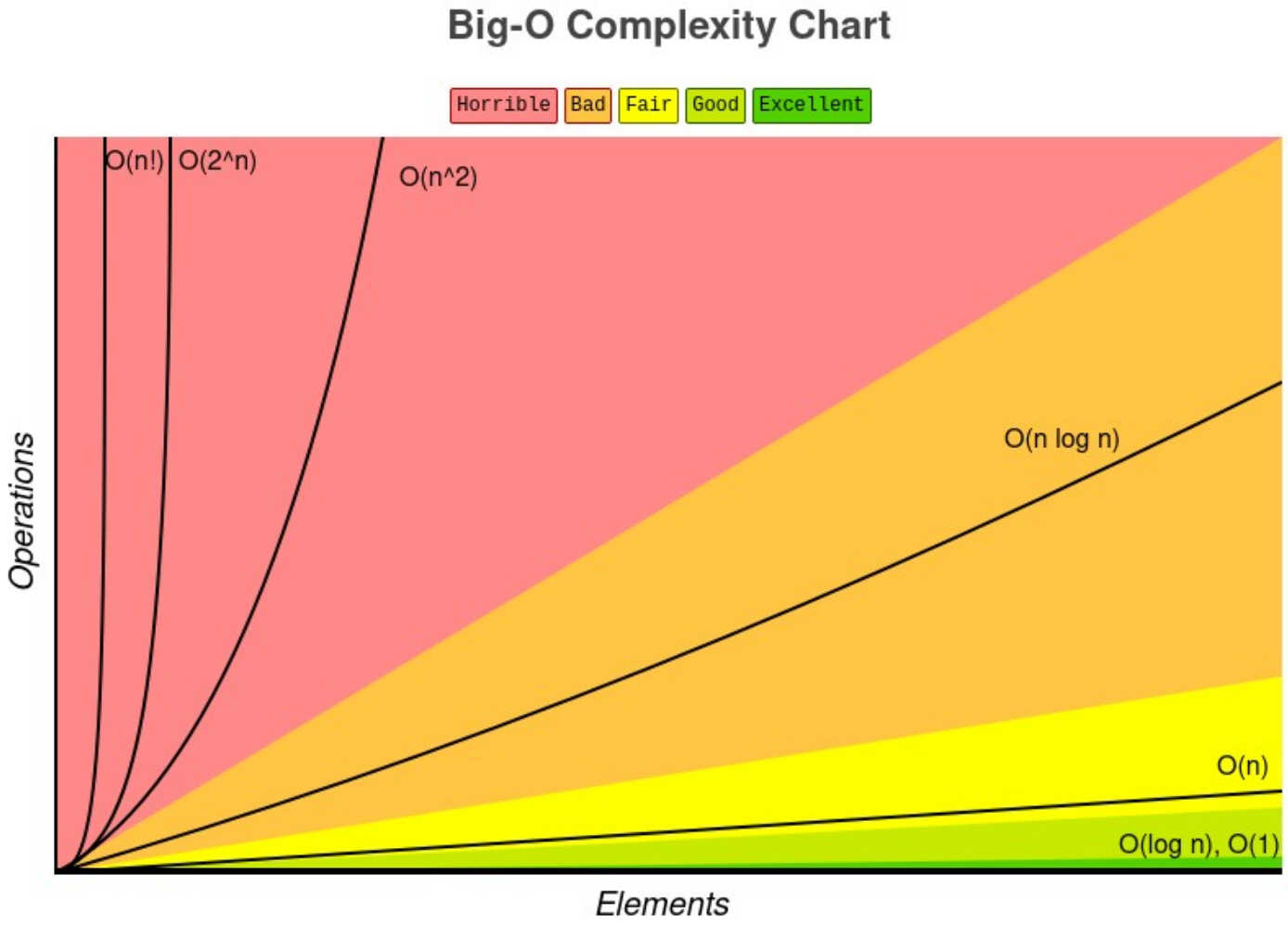
- **Big O**
  - What is it used for?
  - Why is it used?
- **Complexities**
  - What is it?
  - How does it relate to Big O?
- **Complexity Details**
  - O(1)
  - O(n)
  - O(log n)
  - O(n log n)
  - $O(n^c)$
  - $O(2^n)$
  - O(n!)

Big O

- **Used in determining efficiency**
- **Measure of complexity in algorithms**
  - A measure of time
  - A measure of space
  - Does not need to account for hardware
- **Some complexities are better than others**

Big O Complexity

- **Best**
  - O(1)
  - O(log n)
- **Worst**
  - O(n!)
  - O(c^n)
  - O(n^c)



https://www.bigocheatsheet.com/

O(1)

- **Constant complexity**
- **The same regardless of number of items processed**
- **Best overall**
- **Seen in assignments, declarations, and calls**
  - Int a = 0;
  - Void func_1();

O(n)

- **Linear complexity**
- **Every item increases time by a factor of one**
- **An average complexity**
- **Usually seen in loops**
  - for(int I = 0; I < n; I++)
  - while (a < n)

O(log n)

- Logarithmic complexity
- Best without being constant
- Rarer than most
- Most common examples in sorts and searches
- A log n loop

```
Int n = 10;
While(n>0){
    n = n/2;
}
```

O(n log n)

- **Logarithm multiplied by linear**
- **Very average**
  - Worse than O(n)
  - Better than exponential algorithms
- **Also common in search and sort**
- **A nested loop example**

```
Int n = 10;
for(int a = n; a > n; a--)
{
        int b = n;
        while(b>0){
                b = b/2;
        }
}
```

$O(n^c)$

- **Polynomial growth**
- **Terrible complexity**
- **Nested linear loops**
  - C refers to number of loops

```
Int n, m = 10;
While(n>0){
        While(m>0){
                M--;
        }
        N--;
}
```

$O(2^n)$

- **Exponential growth**
- **Almost the worst complexity**
  - Not as bad as factorial growth
- **The tower of Hanoi**

```
void solve_hanoi(int N, string from_peg, string
to_peg, string spare_peg)
{
   if (N<1) {
      return;
   }
   if (N>1) {
      solve_hanoi(N-1, from_peg, spare_peg, to_peg);
   }
   print "move from " + from_peg + " to " + to_peg;
   if (N>1) {
      solve_hanoi(N-1, spare_peg, to_peg, from_peg);
   }
}
```

https://stackoverflow.com/questions/34915869/example-of-big-o-of-2n/34916117

O(n!)

- Factorial growth
- The worst complexity
- Fibonacci functions are a common example

```
unsigned int factorial(unsigned int n)
{
   if (n == 0)
   return 1;
   return n * factorial(n - 1);
}
```

https://www.geeksforgeeks.org/program-for-factorial-of-a-number/

Citations

- https://www.geeksforgeeks.org/analysis-algorithms-big-o-analysis/
- https://www.bigocheatsheet.com/
- https://en.wikipedia.org/wiki/Big_O_notation#Big_Omega_notation
- https://rob-bell.net/2009/06/a-beginners-guide-to-big-o-notation/
- https://yourbasic.org/algorithms/big-o-notation-explained/
- https://www.quora.com/How-can-we-check-for-the-complexity-log-n-and-n-log-n-for-an-algorithm#
- https://www.geeksforgeeks.org/program-for-factorial-of-a-number/