Making Analysis of Algorithms and Data Structures Make Sense

Catherine Leung
School of ICT
Seneca College

Contents

- Review of Big-O notation
- Growth Rates
- Counting operations
- Examples
- Resources

Algorithm and Data Structures

- Ways to store, retrieve, and manipulate data

Analysis

- Measure for resource usage
- Resources:
  - Anything consumed by your program is a resource
  - Top two are time and memory
  - But not only those two

Big-O

- A way to measure resource needs
- Based on amount of data
- Does NOT measure amount of resources needed to processes n pieces of data
- Does measure amount of resource increase to process n+1 pieces of data
- All about growth rates!

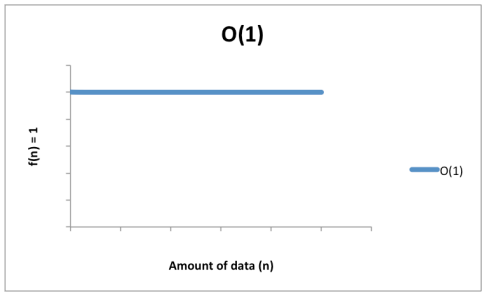
Formally

- Let the function T(n) represent the amount of resources needed to process n pieces of data by a particular algorithm
- T(n) is O(f(n)) iff there exists two constants *c* and n_o such that T(n) <= c f(n) for all n > n_o
- Yikes! How to explain that?

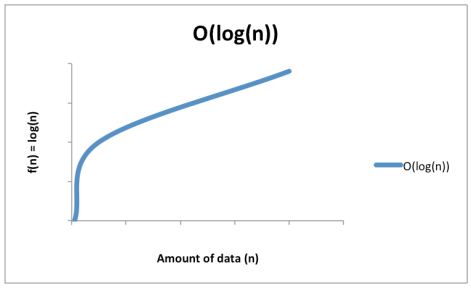
The f(n) inside the O()

- The function f(n) describes a curve

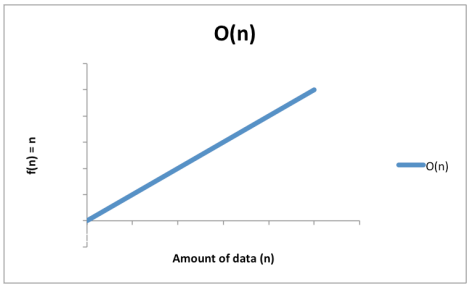
f(n) = 1
constant



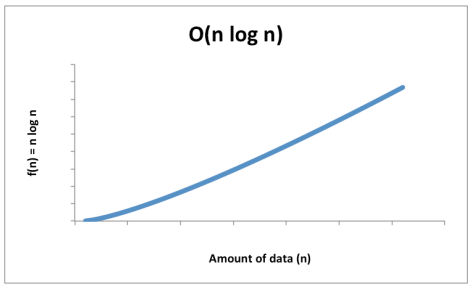
f(n) = log (n)
logarithmic



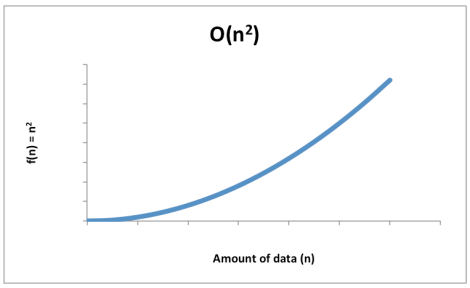
f(n) = n
linear



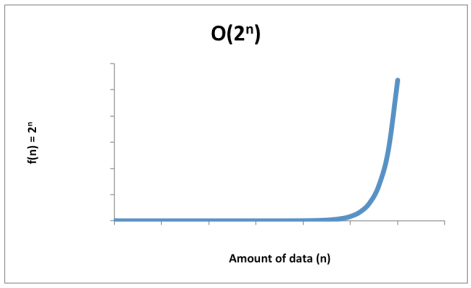
f(n) = n log n



f(n) = n^2
Quadratic



$f(n) = 2^n$



Big-O as a rating system

- O(1)
- O(log n)
- O(n)
- O(n log n)
- O(n^2)
- O(n^3)
- O(2^n)

smaller growth in resource need

larger growth in resource need

The c and n_o

- T(n) is O(f(n)) iff there exists two constants *c* and n_o such that T(n) <= c f(n) for all n > n_o
- If we can describe our resource need with a function T(n) we can give it a "rating", a best fit curve
- The c allows us to stretch the curve
- The n_o allows the statement to not have to be true for all n, n > n_o

Linear Search

```
int linearSearch(int key,int arr[],
                                int size){
      int rc=-1;
      for(int i=0;i<size && rc==-1;i++){
            if(arr[i]==key){
                rc=i;
        }
      }
      return rc;
}
```

More…

- Lots of DS and algs to choose from… please pick:
  - Stacks/Queues (implementation array vs. linked list)
  - Heaps and Heapsort
  - Binary search trees
  - Hash Tables

Resources

- Code/slides from this talk:
  - https://github.com/cathyatseneca/CEMC2015
- My data structures and algorithms notes:
  - https://www.gitbook.com/book/cathyatseneca/data-structures-and-algorithms/details
- My animations:
  - http://cathyatseneca.github.io/DSAnim/index.html