CS202 – Data Structures

LECTURE-03

Asymptotic Analysis

Big-O, Big-Omega, Big-Theta, List ADT

Dr. Maryam Abdul Ghafoor
Assistant Professor
Department of Computer Science, SBASSE

Agenda

- Asymptotic Analysis
- Upper, Lower and Tight Bound

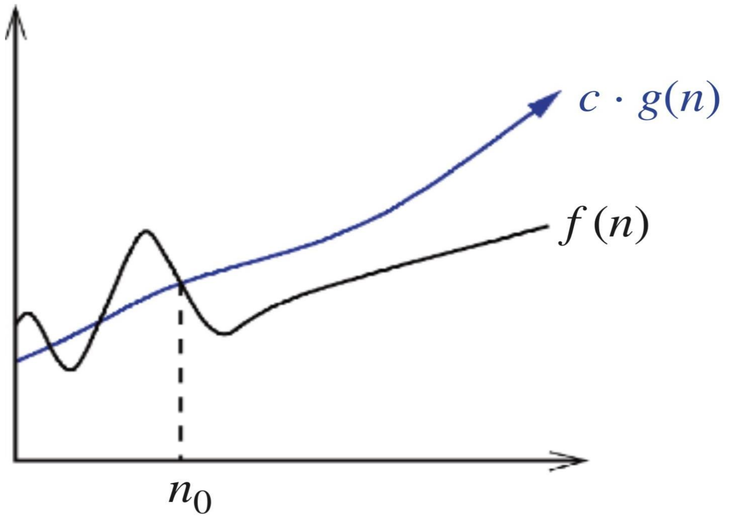
Recap

- Experimental Analysis
  - Actual runtime, accurate
  - Different input size
- Asymptotic Analysis
  - Calculates time by analyzing pseudocode
  - Eliminates the constants and lower order terms
  - Ignore the coefficient of highest order term

Asymptotic Analysis: Big-O

$f(n)$ is in $O(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \leq c.O(g(n))$$



Note: c > 0 and $n_0 \geq 1$ (their values must NOT depend on n)

Informally, this means is $f(n)$ is less than some constant multiple of $g(n)$

Complexity Analysis

```
void doSomething(int arr[], int n) {
 bool swapped;
 for (int i = 0; i < n - 1; i++) {
     for (int j = 0; j < n - i - 1; j++)
     {
        if (arr[j] > arr[j + 1]) {
           int temp = arr[j];
           arr[j] = arr[j + 1];
           arr[j + 1] = temp;
         }
     }
  }
}
```

What does this algorithm do?

Estimate the time complexity of this algorithm in the terms of O(.)

Compute the exact number of steps this algorithm will perform in the worst case on an input of size n. (take-home task)

Let's do Some Practice!

- Is $10n + 5 \in O(n)$?

$f(n)$ is $O(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \leq c.O(g(n))$$

Solve the inequality for c to find any (c, n_o) pair of values that satisfy the definition

Analyzing Complexities

```
int getFirst(int arr[], int n) {
    return arr[0];
}
```

O(1)

```
int sumArray(int arr[], int n) {
    int sum = 0;
    for (int i = 0; i < n; i++)
        sum += arr[i];
    return sum;
}
```

O(n)

Analyzing Complexities

```
int mystry(int arr[], int n, int key) {
    int low = 0, high = n - 1;
    while (low <= high) {
        int mid = (low + high) / 2;
        if (arr[mid] == key)
            return mid;
        else if (arr[mid] < key)
            low = mid + 1;
        else high = mid - 1;
    }
    return -1;
}
```

O(log n)

Analyzing Complexities

```
int find(int x, int y, int n) {
    if (x < 0 || y < 0 || x >= n || y >= n)
        return 0;
    if (x == n-1 && y == n-1)
        return 1;
    return find(x+1, y, n) + find(x-1, y, n)
        + find(x, y+1, n) + find(x, y-1, n);
}
```

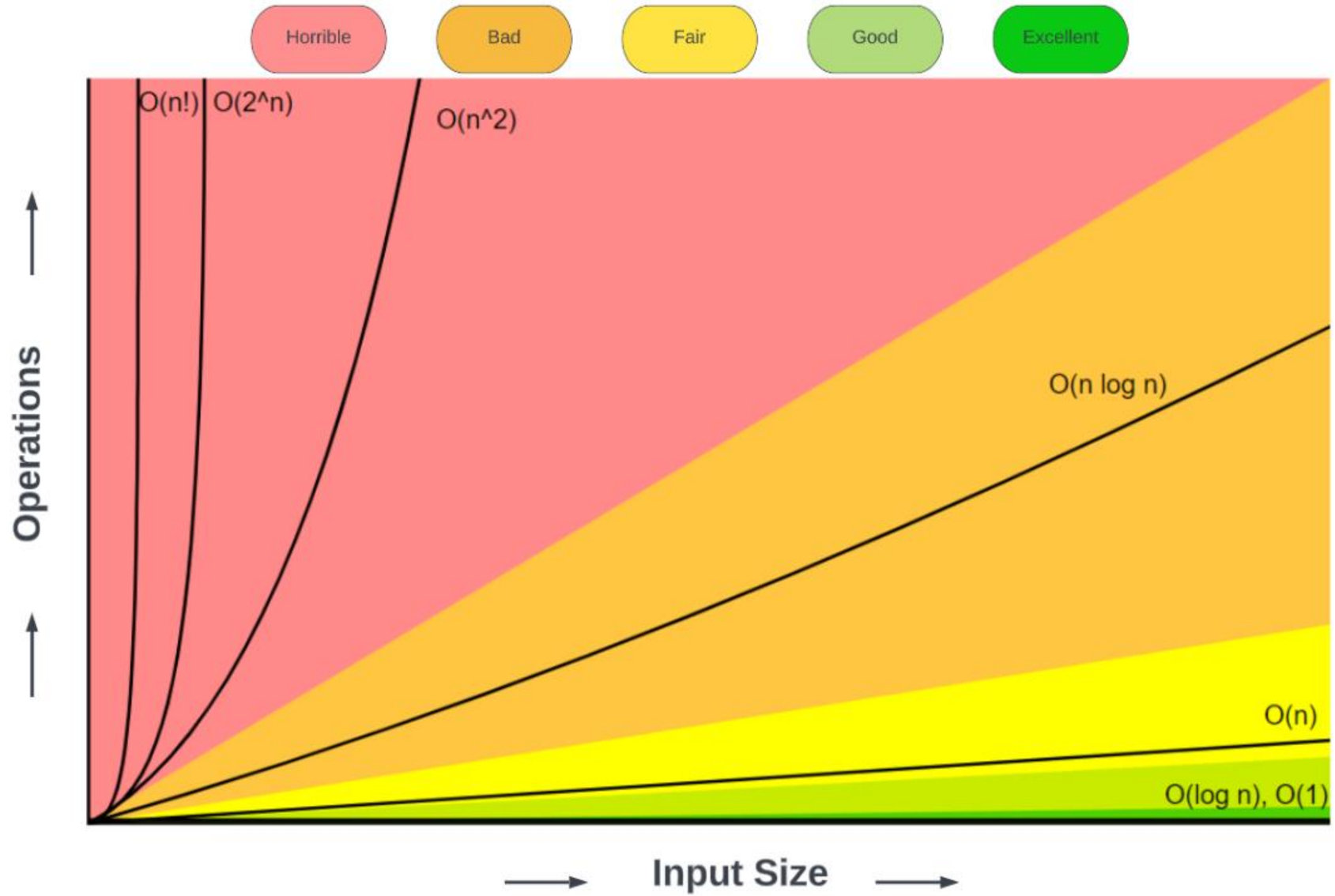
$O(4^n)$

Analyzing Complexities

```
void permute(string s, int l, int r) {
    if (l == r) {
        cout << s << endl;
        return;
    }
    for (int i = l; i <= r; i++) {
        swap(s[l], s[i]);
        permute(s, l + 1, r);
        swap(s[l], s[i]); // backtrack
    }
}
```

O(n!)

Big-O Complexity Chart for Common Functions



https://www.bigocheatsheet.com

Let's do Some Practice!

- Is $2n \in O(n^2)$?

$f(n)$ is $O(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \leq c.O(g(n))$$

Solve the inequality for c to find any (c, n_o) pair of values that satisfy the definition

Let's do Some Practice!

- Is $n^2 \in O(n)$?

$f(n)$ is $O(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \leq c.O(g(n))$$

Solve the inequality for c to find any (c, n_o) pair of values that satisfy the definition

Let's do Some Practice!

- Is $e^{3n} \in O(e^n)$?

$f(n)$ is $O(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \leq c.O(g(n))$$

$$e^{3n} \leq c \cdot e^n$$

$$c \geq \frac{e^{3n}}{e^n}$$

$$c \geq e^{2n}$$

Solve the inequality for c to find any (c, n_o) pair of values that satisfy the definition

Concept Check!

Poll

What is the O(.) for this code?

1. O(n)
2. $O(n^2)$
3. $O(n+n^2)$
4. Other/none

Algorithm 1 Linear search algorithm

```
function LINEAR-SEARCH(a[ ],size,key)
    for i = 1 to n do
        if a[i] = key then
            return true
    return false
```

Concept Check!

Poll

```
for(int i=1;i<n;i++){
  for(int j=1; j < 2n;j*=2){
    count +=1;
  }
 }
```

What is the upper bound O() for this code?

O(n)

$O(n^2)$

$O(n^3)$

O(nlogn)

$O(n\log n^2)$

Big-O Analysis - Summary

- Big-O simplifies analysis by focusing on the most dominating term

Number of Primitive Operations(Best Case)

Activity

	Statements	Steps
1	`int find(vector<int>& grades, int size, int f){`	
2	`  // size = n, f = numberToFind`	
3	`  for ( int i = 0; i < n; i++)`	
4	`  if ( grades[i] == f)`	
5	`  return i;`	
6	`  return -1;`	
7	`}`	
Total		

Number of Primitive Operations(Best Case)

	Statements	Steps
1	`int find(vector<int>& grades, int size, int f){`	
2	`  // size = n, f = numberToFind`	
3	`  for ( int i = 0; i < n; i++)`	2
4	`  if ( grades[i] == f)`	2
5	`  return i;`	1
6	`  return -1;`	
7	`}`	
Total		5

Number of Primitive Operations(Worst Case)

	Statements	Steps
1	`int find(vector<int>& grades, int size, int f){`	
2	`  // size = n, f = numberToFind`	
3	`  for ( int i = 0; i < n; i++)`	1 + (n+1) + 2n
4	`  if ( grades[i] == f)`	2n
5	`  return i;`	1
6	`  return -1;`	1
7	`}`	
Total		5n + 3

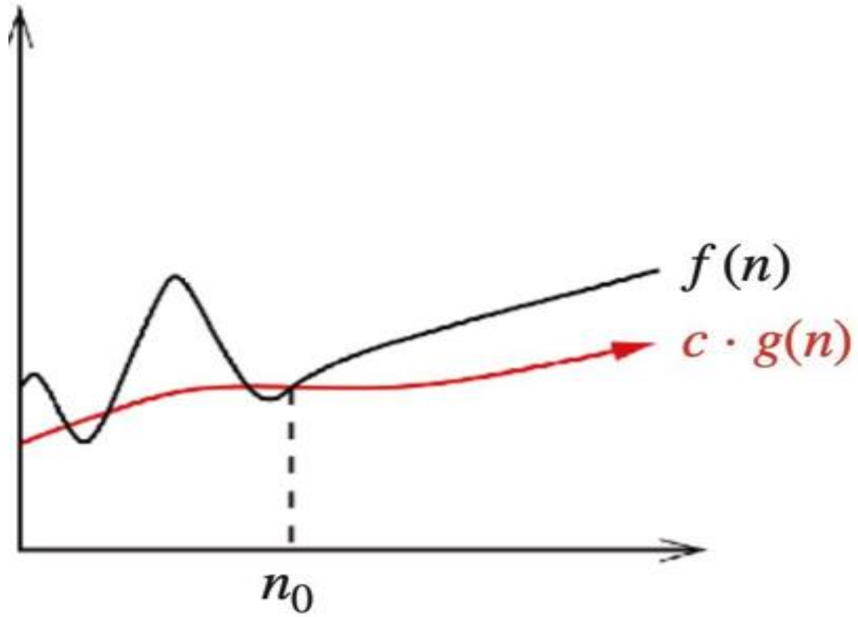
Other Asymptotic Notations

- Big-Omega: Ω(.)
- Big-Theta: Θ(.)

Asymptotic Analysis: Big-Omega

$f(n)$ is in $\Omega(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \geq c.\Omega(g(n))$$



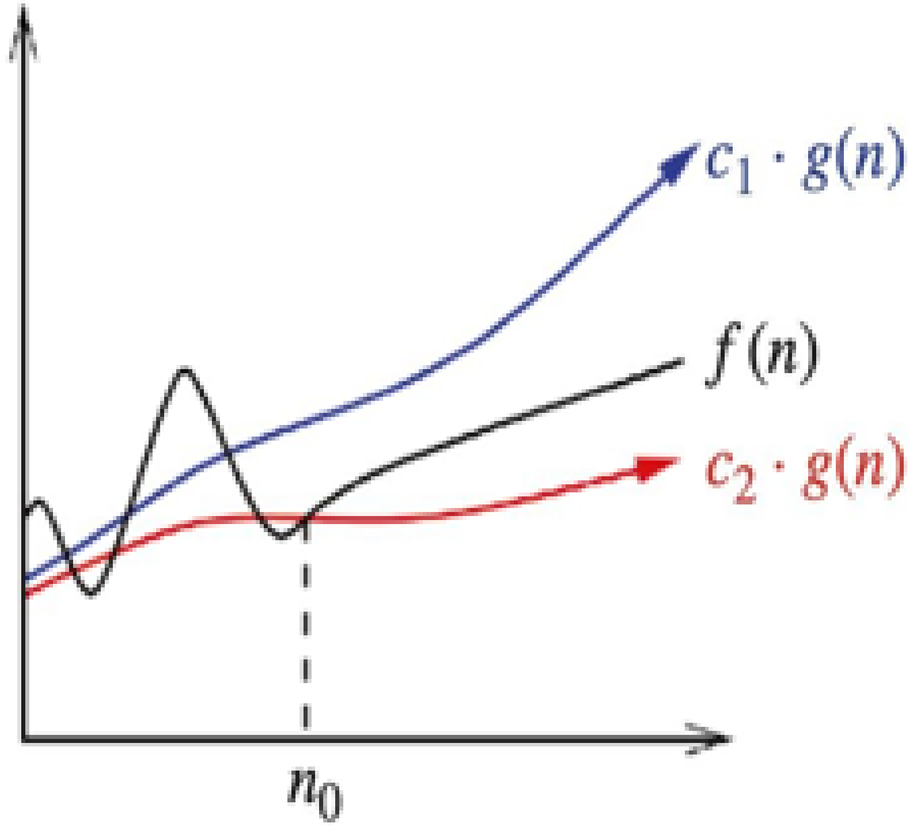
Note: c > 0 and $n_0 \geq 1$ (their values must NOT depend on n)

Informally, this means is $f(n)$ is greater than some constant multiple of g(n)

Asymptotic Analysis: Big-Theta

$f(n)$ is in $\Theta(g(n))$

If $f(n)$ *is in* $O(g(n))$ *and* $f(n)$ *is in* $\Omega(g(n))$



Let's Practice

- Is $10n^2 \in \Omega(n^2)$?

$f(n)$ is in $\Omega(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \geq c.\Omega(g(n))$$

Let's Practice

$8n^3 + 20n^2 + 50n + 100 \in \Omega(n)$

$f(n)$ is in $\Omega(g(n))$ if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \geq c.\Omega(g(n))$$

Let's Practice

Is $5n^2 \in \theta(n)$?

$f(n)$ is in $\Theta(g(n))$, if there exists positive constants c, n_o such that for all $n \geq n_o$

$$f(n) \geq c.\mathrm{O}(g(n))$$

and

$$f(n) \geq c.\Omega(g(n))$$

Questions