CSE12 - Lecture 11

Wednesday, April 26, 2023 8:00 AM

PA3 → due today @ 10pm

PA4 → released today → due Tuesday

Exam 1 → Friday

↳ BFS/DFS and before

Measuring Runtime

Count how many times each line executes, then say which Θ( ) statement(s) is(are) true.

```
int maxDifference(int[] arr){
   max = 0;
   for (int i=0; i<arr.length;  i++) {
      for (int j=0; j<arr.length; j++) {
         if (arr[i] - arr[j] > max)
            max = arr[i] - arr[j];
      }
   }
   return max;
}
```

1

$1 + (n+1) + n$

$n(1 + (n+1) + n$

N

0 or N

0 best case N worst case

$3 + 2n + n(2 + 4n)$

Assume n = arr.length

$f(n) = 4n^2 + 4n + 3$

$4n^2 + 4n^2 + 3$

$8n^2 + 3$

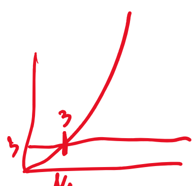
$c = 8$

$N_0 = 3$

$g(n) = n^2$

$\Theta(n^2)$

A. $f(n) = \theta(2^n)$
B. $f(n) = \theta(n^2)$
C. $f(n) = \theta(n)$
D. $f(n) = \theta(n^3)$
E. Other/none/more



Big O ___ upper bound

$f(n) =$ O $(g(n))$, $f(n) <= c * g(n)$
for all $n \geq n0$

Big Ω omega lower bound

$f(n) =$ Ω $(g(n))$, $f(n) >= c * g(n)$
for all $n \geq n0$

Big Θ theta tight bound

$f(n) =$ Θ $(g(n))$, $f(n) = c * g(n)$
for all $n \geq n0$

For each function in the list below, it is related to the function below it by O, and the reverse is not true. That is, n is $O(n^2)$ but n^2 is not $O(n)$.

- $f(n) = 1/(n^2)$
- $f(n) = 1/n$
- $f(n) = 1$
- $f(n) = \log(n)$
- $f(n) = \text{sqrt}(n)$
- $f(n) = n$
- $f(n) = n^2$
- $f(n) = n^3$
- $f(n) = n^4$
- … and so on for constant polynomials …
- $f(n) = 2^n$
- $f(n) = n!$
- $f(n) = n^n$

$n = 1000$
1000 000

$n = 10000$
100000000

Count how many times each line executes, then say which Θ( ) statement(s) is(are) true.

```
int sumTheMiddle(int[] arr){
   int range = 100;
   int start = arr.length/2 - range/2;
   int sum = 0;
   for (int i=start; i<start+range;  i++)
   {
      sum += arr[i];
   }
   return max;
}
```

1
1
1
$1 + 101 + 100$
100
1

range = 100

start = $\frac{N}{2} - 50$

start + range = $\frac{N}{2} + 50$

Assume n = arr.length

A. $f(n) = \theta(2^n)$
B. $f(n) = \theta(n^2)$
C. $f(n) = \theta(n)$
D. $f(n) = \theta(1)$
E. None of these

constant time

$N = 100$
start $= \frac{100}{2} - 50 = 0$
start + range $= \frac{100}{2} + 50 = 100$
→ 100

$N = 10000$
start $= \frac{10000}{2} - 50 = 4950$
start + range $= \frac{10000}{2} + 50 = 5050$
100

$f(n) = 306$
$c = 306$
$N_0 = 0$

$g(n) = 1$
$\Theta(1)$

Name: ______________ PID: ______________ Code: 4020

```
void printAllItemsTwice(int arr[], int size)
{
    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }

    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }
}
```

N + N → $2N$

$1 + (N+1) + N$ $3N+2$

$1 + (N+1) + N$ $+ \; 3N+2$

$6N+4$ $g(N) = N$

$c = 6$

$N_0 = 4$ $\Theta(N)$

What is the tight bound?

$\Theta(N)$

```
void printFirstItemThenFirstHalfThenSayHi100Times(int arr[], int size)
{
    printf("First element of array = %d\n",arr[0]);

    for (int i = 0; i < size/2; i++) {
        printf("%d\n", arr[i]);
    }

    for (int i = 0; i < 100; i++) {
        printf("Hi\n");
    }
}
```

1

$\frac{N}{2}$ $1 + (\frac{N}{2}+1) + \frac{N}{2}$ $\frac{3N}{2}+2$

100 $1 + 101 + 100$ 302

$f(N) = \frac{3N}{2} + \cancel{304}$

$\Theta(N)$

What is the tight bound?

$\Theta(N)$

```
void printAllNumbersThenAllPairSums(int arr[], int size)
{
    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }

    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            printf("%d\n", arr[i] + arr[j]);
        }
    }
}
```

N $1 + (N+1) + N$ $3N+2$

$+$

N N $1 + (N+1) + N$ $2N+2 \; +$

$N(1 + (N+1) + N)$ $3N^2 + 2N$

$f(N) = 3N^2 + 7N + 4$

$\Theta(N^2)$

What is the tight bound?

N^2+2 → $\Theta(N^2)$

Selection Sort

```
import java.util.Arrays;
public class Sort {
   public static void sortA(int[] arr) {
      for(int i = 0; i < arr.length; i += 1) {
         System.out.print(Arrays.toString(arr) + " -> ");
         int minIndex = i;
         for(int j = i; j < arr.length; j += 1) {
            if(arr[minIndex] > arr[j]) { minIndex = j; }
         }
         int temp = arr[i];
         arr[i] = arr[minIndex];
         arr[minIndex] = temp;
         System.out.println(Arrays.toString(arr));
      }
   }
}
```

N

$\frac{N}{2}$

Selection Sort – what does it print out?

N=5

Sort.sortA(new int[]{ 53, 83, 15, 45, 49 });

[53, 83, 15, 45, 49] -> 15 | 83 53 45 49   5

15 45 | 53 83 49   4

15 45 49 | 83 53   3

15 45 49 53 | 83   2

15 45 49 53 83 | 1

$\frac{N}{2}$

worse case: reverse sorted array

83 53 49 45 15

best case: sorted array

15 45 49 53 83

What is the runtime? Consider the shape of the input array.

Worse case: $\Theta(N^2)$

Best case: $\Theta(N^2)$ → isSorted() $\Theta(N)$

Insertion Sort

```
import java.util.Arrays;
public class Sort {
   public static void sortB(int[] arr) {
      for(int i = 0; i < arr.length; i += 1) {
         System.out.print(Arrays.toString(arr) + " -> ");
         for(int j = i; j > 0; j -= 1) {
            if(arr[j] < arr[j-1]) {
               int temp = arr[j-1];
               arr[j-1] = arr[j];
               arr[j] = temp;
            }
         }
         System.out.println(Arrays.toString(arr));
      }
   }
}
```

N

$\frac{N}{2}$

$\frac{N}{2}$

Insertion Sort – what does it print out?

```
Sort.sortB(new int[]{ 53, 83, 15, 45, 49 });

[53, 83, 15, 45, 49] ->
```

53 83 15 45 49

53 83 15 45 49

15 53 83 45 49

15 45 53 83 49

15 45 49 53 83

What is the runtime? Consider the shape of the input array.

Worse case: $\Theta(N^2)$

Best case: $\Theta(N^2)$