CSC1001 Tutorial 10

Algorithm Analysis

Frederick Khasanto (122040014)

16 November 2023

Primitive operations

---> $O(1)$

- Assigning a variable to an object (`a = 5`)
- Determining the object associated with a variable (`b = a + 5`)
- Performing an arithmetic operation (`+,-,*,/,%`)
- Comparing two numbers (`a > b`)
- Accessing an element in array/list by index (`b = a[1]`)
- Calling a function (`func()`)
- Returning from a function (`return`)

Big-Oh Notation

Definition

$f(n)=O(g(n))$ if $f(n)\leq c\,g(n)$ for $n\geq n_0$

Order

$c<log\,n<n<n\,log\,n<n^2<n^3<a^n$

Rules

1. Polynomial Rule --> *Biggest power eats all*

   If $g(n)$ is a non-negative polynomial of highest degree d, then $g(n)=O(n^d)$

   e.g., $7n^4+5n^3+3n^2=O(n^4)$

1. Product rule --> *Big multiplies the big*

   $g_1(n)=O(f_1(n)), g_2(n)=O(f_2(n))$

   then, $g_1(n).g_2(n)=O(f_1(n).f_2(n))$

e.g., $(n^5+n^2+1)(n^4+n^{13}+n^2)=O(n^{18})$

1. Sum rule --> *Biggest of the two big*

   $g_1(n)=O(f_1(n)), g_2(n)=O(f_2(n))$

   then, $g_1(n)+g_2(n)=O(max(f_1(n), f_2(n)))$

   e.g., $g(n)=(5n^3+3n)+(7n^4+2n^5)=O(max(n^3,n^5))=O(n^5)$

1. Log rule --> *Log only beats constant*

   e.g., $g(n)=(\log n)^2+n^{0.0001}=O(n^{0.0001})$

1. Exponential rule --> *Exponential beats powers*

   $g(n)=a^n+n^b$, where $a>1 \Rightarrow g(n)=O(a^n)$

   $g(n)=a^n+b^n$, where $a>b>1 \Rightarrow g(n)=O(a^n)$

   e.g., $g(n)=2^n+n^2=O(2^n)$

   $g(n)=2^n+5^n=O(5^n)$

Questions

Q1: Verify $8n\log n$ better than $2n^2$

The number of operations executed by algorithms A and B is $8n\log n$ and $2n^2$, respectively. Determine n_0 such that A is better than B for $n \geq n_0$

Find the intersection between the two functions. They meet at $n=n_0$.

$8n_0\log_2 n_0=2{n_0}^2$

$\log_2 n_0=\frac{1}{4}n_0$

$n_0=16$

For $n \geq 16, 8n\log n<2n^2$.

Q2: Verify $40n^2$ better than $2n^3$

The number of operations executed by algorithms A and B is $40n^2$ and $2n^3$, respectively. Determine n_0 such that A is better than B for $n \geq n_0$

Find the intersection between the two functions. They meet at $n = n_0$.

$40 {n_0}^2 = 2 {n_0}^3$

$n_0 = 20$

For $n \geq 20, 40n^2 < 2n^3$.

Q3: Ordering function asymptotically

Best to worst:

$2^{\log n}, 2^{10}, 4n, 3n + 100\log n, n\log n, 4n\log n + 2n, n^2 + 10n, n^3, 2^n$

Q4: Prove Big-Oh of a function

i) Show that $8n+5$ is $O(n)$

$8n + 5 \leq c\,n$ for $n \geq n_0$

Let $c = 9$, $8n + 5 \leq 9n \Rightarrow n \geq 5$

Therefore, $8n + 5 = O(n)$

ii) Show that $5n^4 + 3n^3 + 2n^2 + 4n + 1$ is $O(n^4)$

$5n^4 + 3n^3 + 2n^2 + 4n + 1 \leq c n^4$ for $n \geq n_0$

Since $n^3, n^2, n^1, n^0 \leq n^4$ for all $n \geq 1$, then

$LHS \leq 5n^4 + 3n^4 + 2n^4 + 4n^4 + n^4 = 15n^4$

Let $c = 15$, $LHS \leq c n^4$ for all $n \geq 1$

Therefore, $5n^4 + 3n^3 + 2n^2 + 4n + 1 = O(n^4)$

iii) Show that $5n^2 + 3n\log n + 2n + 5$ is $O(n^2)$

Since $\log n \leq n$ for all $n \geq 1$, then

$LHS \leq 5n^2 + 3n*n + 2n^2 + 5n^2 = 15n^2$

Let $c = 15$, $LHS \leq cn^2$ for all $n \geq 1$

Therefore, $5n^2 + 3n\log n + 2n + 5 = O(n^2)$

iv) Show that $16n\log n + n$ is $O(n\log n)$

Since $n \leq n\log n$ for all $n \geq 2$

$16n\log n + n \leq 16n\log n + n\log n = 17n\log n$

Let $c = 17$, $LHS \leq cn\log n$ for all $n \geq 2$

Therefore, $16n\log n + n = O(n\log n)$

Q5: Time Complexity

What is the time complexity of `fun(n)`?

```python
def fun(n):
    count = 0
    m = n//2
    for i in range(n, 0, -m):
        m=m//2
        for j in range(0, i, 1):
            print('i is %d, j is %d'%(i, j))
            count += 1
    print(count)
    return count

# Try running with n = 10, 100, 1000 and see the final printed result
fun(10)
```

```
i is 10, j is 0
i is 10, j is 1
i is 10, j is 2
i is 10, j is 3
i is 10, j is 4
i is 10, j is 5
i is 10, j is 6
i is 10, j is 7
i is 10, j is 8
i is 10, j is 9
i is 5, j is 0
i is 5, j is 1
i is 5, j is 2
i is 5, j is 3
```

```
i is 5, j is 4
15

15
```

----> $O(n)$

Q6: Time Complexity

i) What is the functionality of the function `product()`?

ii) What is the time complexity of this function?

```python
def product(n, m, count):
    if n==0:
        count+=1
        print('Count is: ', count)
        return 0
    elif n==1:
        print('Count is: ', count)
        count+=1
        return m
    else:
        if n%2==1:
            count+=1
            print('n is: ',n)
            return product(n//2, m, count)*2+m  # Input is divided by 
2 every time the function is called.
        else:
            count+=1
            print('n is: ',n)
            return product(n//2, m, count)*2

print('Product is: ',product(8,4,0))
print('Product is: ',product(64,4,0))
print('Product is: ',product(256,4,0))
```

```
n is:  8
n is:  4
n is:  2
Count is:  3
Prodcut is:  32
n is:  64
n is:  32
n is:  16
n is:  8
n is:  4
n is:  2
```

```
Count is:  6
Prodcut is:  256
n is:  256
n is:  128
n is:  64
n is:  32
n is:  16
n is:  8
n is:  4
n is:  2
Count is:  8
Prodcut is:  1024
```

i) The functionality of the function `product(n,m)` is to calculate the product of n and m, that is $n * m$.

ii) The time complexity of this function is $O(\log n)$.