Assignment 1

Tuesday, September 27, 2022 12:08 AM

1. (3 marks) Use the definition of "big Oh" to prove that $\frac{2}{n}$ is $O(n)$.

Find constants $c>0$ and $n_0 \geq 1$ such that:

$\frac{2}{n} \leq cn$

Simplify: $\frac{2}{n} \leq cn$

$2 \leq cn^2$

$\frac{2}{2} \leq \frac{2n^2}{2}$ ← let $c=2$

$1 \leq n^2$

$1 \leq n$

We have found constants $c=2$ & $n_0=1$ that makes the inequality true $\therefore \frac{2}{n}$ is $O(n)$

2. (3 marks) Let $f(n)$ and $g(n)$ be positive functions such that $f(n)$ is $O(g(n))$. Use the definition of "big Oh" to prove that $f(n) \times g(n)$ is $O(g^2(n))$, where $g^2(n) = g(n) \times g(n)$.

Since $f(n)$ is $O(g(n))$ $\therefore$ by the definition of "big O", there are constants $c>0$ & $n_0 \geq 1$, such that:

$f(n) \leq c \times g(n)$

We need to find constants $c>0$ and $n_0 \geq 1$ such that

$f(n) \times g(n) \leq c \times g^2(n)$

$f(n) \times g(n) \leq c \times g^2(n)$

$\frac{f(n) \times g(n)}{g(n)} \leq \frac{c \times g(n) \times g(n)}{g(n)}$] $g(n)$ is positive $\therefore g(n) > 0$

$\boxed{f(n) \leq c \times g(n)}$

Earlier I demonstrated the there are constants $c>0$ & $n_0 \geq 1$ such that $f(n) \leq c \cdot g(n)$.

Since $f(n) \cdot g(n) \leq c \times g^2(n)$ simplifies to $f(n) \leq c \times g(n)$, we know that there exists constants $c>0$ & $n_0 \geq 1$ such that:

$f(n) \times g(n) \leq c \times g^2(n)$

3. (3 marks) Use the definition of "big Oh" to prove that $n^3 + \frac{n^4}{4}$ is not $O(n^3)$

Assume that the claim is false and $n^3 + \frac{n^4}{4} \in O(n^3)$.

That means we can find constants $c > 0$ and $n_0 \geq 1$ such that:

$$n^3 + \frac{n^4}{4} \leq c \times n^3$$

Simplify: $\frac{n^3(1 + \frac{n}{4})}{n^3} \leq \frac{c \times n^3}{n^3}$ $\Big]$ $n \geq n_0 \geq 1$

$$1 + \frac{n}{4} \leq c$$
$$n \leq 4(c-1)$$

The inequality $n \leq 4(c-1)$ is only valid for values of n that are at most $4(c-1)$, meaning that this constant cannot be true for all values of n larger than some constant n_0.

Specifically if we choose $n \geq 4(c-1) + n_0$, these values of n are larger than or equal to n_0 but not at most $4(c-1)$.

Therefore we have reached a contradiction as there are no constant values $c > 0$ & $n_0 \geq 1$ such that $n^3 + \frac{n^4}{4} \leq c \times n^3$ for all $n \geq n_0$. Therefor $n^3 + \frac{n^4}{4}$ is not $O(n^3)$.

4. The Algorithm is correct.

Termination:

If $x = L[i]$ the algorithm terminates by returning i.
If $x \neq L[i]$ the algorithm changes the value stored in $L[i]$ to be -1.
In the case that the array does not contain x, all the values in the array will become -1 as the algorithm traverses the array from beginning to end. Once all the values in the array of length n have become -1, the while loop will increment i until $i = n$ because all values for $L[i]$ will equal -1. Since $i = n$ the algorithm will return -1 to terminate. After a finite number of recursive calls and loop iterations, all the values in the array will equal -1 and the algorithm ends.

Correctness;

Case 1: x is in L

The algorithm will check each position of the array for the x value by incrementing the i-value using the while loop. It will not pass the while condition the first time through because the first value stored in $L[i] \neq -1$.

As discussed before, the algorithm terminates by either returning i or -1. Since "return -1" is only executed when x is not in L, the algorithm will return i with:

else if $L[i] = x$ then return i

Case 2: x is not in L

In each recursive call, the algorithm replaces one value that is different from x, with -1. The algorithm "checks" each value stored in L. Since all values will eventually equal -1, the algorithm will correctly return -1 through a 2 step process.

1. While loop iterates until $i=n$ because all $L[i]=-1$.
2. Return -1 with this line:
   if $i=n$ then return -1.

5.

c=39 x=0	target=40	ret addr=A3	ret val= 0	
target=39 c=30 x=1	m=40	ret addr=A2	ret val=	
target=40 c=20 x=1	m=30	ret addr= A1	ret val=	
pos=20	res=	ret addr=05		

6

n	Linear Search
5	309
10	378
100	731
1000	4899
10000	11796
100000	35678

n	Quadratic Search
5	545
10	1037
100	10762
1000	176399
10000	12647422

n	Factorial Search
7	3078400
8	16189401
9	88551900
10	1189027601
11	9590898800
12	79721832900