CSE548/AMS542 Fall 2013 Analysis of Algorithms

Homework 1

Himanshu Shah
ID : 109324380

Himanshu Shah
109324380

Homework 1

Problem 1:

A) Ch-2 Problem 4 [KT]:

Take the following list of functions and arrange them in ascending order of growth rate. That is, if function g(n) immediately follows function f(n) in your list, then it should be the case that f(n) is O(g(n)).

$(a)g_1(x) = 2^{\sqrt{\log n}}$

$(b)g_2(x) = 2^n$

$(c)g_4(x) = n^{4/3}$

$(d)g_3(x) = n(\log n)^3$

$(e)g_5(x) = n^{\log n}$

$(f)g_6(x) = 2^{2^n}$

$(g)g_7(x) = 2^{n^2}$

Solution:

It can be easily noted that $g_7(x) = O(g_6(x))$ and $g_2(x) = O(g_7(x))$. Also on taking the value of n=4, 16 and 64 the following order of growth can be seen:

$g_1(x) = O(g_4(x))$

$g_4(x) = O(g_3(x))$

$g_3(x) = O(g_5(x))$

$g_5(x) = O(g_2(x))$

Hence the growth of the functions in increasing order is :

$g_1(x),\ g_4(x),\ g_3(x),\ g_5(x),\ g_2(x),\ g_7(x)$ and $g_6(x)$. With $g_1(x)$ being the lowest and $g_6(x)$ being the largest

B) Ch-2 Problem 5 [KT]:

Assume you have functions f and g such that f(n) is O(g(n)). For each of the following statements, decide whether you think it is true or false and give a proof or counterexample.

(i) $\log_2 f(n)$ **is** $O(\log_2 g(n))$

Solution:

True

- This is evident from the fact that Log is an increasing function. Hence if $x > y$ then $\log_2(x)$ will be greater than $\log_2(y)$.

f(n) is O(g(n)) so, let f(n) = x^2 and g(n) = x^3.

Now $x^2 \leq cx^3$
Applying $\log_2$ on both sides
$\log_2(x^2) \leq c\log_2(x^3)$
Hence, proved.
(ii) $2^{f(n)}$ is $O(2^{g(n)})$
-True
The proof for this is pretty obvious, Considering the fact that exponent function is an increasing function.
f(n) is O(g(n)) so, let f(n) = x^2 and g(n) = x^3.

So, $x^2 \leq cx^3$

Consider LHS = 2^{x^2} and RHS = 2^{x^3} multiplied by some constant m.

as $x^2 \leq cx^3$
$2^{x^2} \leq m\,2^{x^3}$
Hence $2^{f(n)}$is $O(2^{g(n)})$.
Thus, proved.
(iii) $f(n)^2$ is $O(g(n)^2)$
- True
- Here we will prove using the property of Big-Oh function.
According to the rule of Big-Oh multiplication by a function,
$O(f(n))O(g(n)) = O(f(n)g(n))$

So, as $f(n)$is$O(g(n))$
if we multiple $f(n)$ with $f(n)$ we get a
$f(n)^2 = O(g(n)g(n))$
$f(n)^2 = O(g(n)^2)$

C) Ch-2 Problem 8 [KT]:

Solution a)
This problem can be solved using the following strategy:
Try dropping the jar at every even number of rugs starting from level L = 2 and then increasing to $L_{\text{new}} = L_{\text{current}} + 2$, until, the jar breaks.
Go one level lower $L_{\text{new}}=L_{\text{current}} - 1$, if the jar breaks, its highest safe rug level is $L_{\text{current}} - 1$.

Else it is $L_{current}$.

Example suppose a jar has the safe level of 6. Then, as per the strategy, one should try dropping it at level 2 , 4 ,6 and then 8. The jar will break on 8. So try level 7, and it will break again and hence proving that 6 is its highest safe rug level.

Solution b)

Solution for question b is similar to the first one. Infact the solution a is a special case of solution b where number of jars =2.
For this problem, I am considering k number of jars that can be used at maximum and $f_k(x)$ is the number of times an attempt is made. Here, n is the total number of rugs in the ladder.

The strategy is as follows.

Step-1 Based on k, test the strength of the jar at m * k level starting m from 1 and incrementing it by one until the jar breaks.

Step-2 Test the strength at one level less than the previous height and see if it breaks. continue until it stops breaking.

This level is the highest safest level for the jar.

> This algorithm will grow asymptotically lower than as the value of k will increase.

For k=3, n=10 and the highest safest level =7, this algorithm will work as:
> First attempt : 3
> Second attempt : 6
> Third attempt : 9 - breaks
Fourth attempt : 8 - breaks
Fifth attempt : 7 - does not break - SOLUTION Found

For k=4, n=10 and the highest safest level =7, this algorithm will work as:
> First attempt : 4
> Second attempt : 8 - breaks
Third attempt : 7 - does not break - SOLUTION Found

Problem 2. Prove or disprove (i.e., give counter examples) for the following claims. f (n), g(n) are non- negative functions.
$(a)\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

$(b) o(f(n)) \cap \omega(f(n)) = \emptyset$

$(c)(n + a)b = \Theta(\mathrm{n\ b})$, a, b are positive integers

$(d) f(n) = O(f(n)^2)$

$(e) f(n) = O(g(n))$ implies that $2f(n) = O(2g(n))$

Problem 3:

Himanshu Shah
109324380

Problem 4:

Given n unsorted numbers, compute the maximum and minimum using $\lceil 3n/2 \rceil - 2$ comparisons.

Solution:

The algorithm to compute the maximum and minimum on n elements in $\lceil 3n/2 \rceil - 2$ comparisons can be done as follows :

Step -1 :
Compare each of the even numbered element with the consecutive element in the list and determine the one that is greater. Add the elements that are larger in a set and name it SET A and the ones that are smaller in another SET B.

Step -2 :
For each element in SET A find the maximum by performing a linear search.

Step - 3:
For each element in SET B find the minimum by performing a linear search.

At the end of the algorithm, we will be able to determine the maximum and minimum from a given set of n numbers.

Calculating the number of comparisons made in each step of the algorithm:

Step -1 : For a given set of n numbers in order to generate two sets for larger and smaller numbers, one needs to make $\lfloor n/2 \rfloor$ Comparisons.

Step -2 : In order to find the maximum from SET A using linear searching technique, one needs to perform, $\lfloor n/2 \rfloor$ -1 comparisons. This is because, the number of elements in the set is $\lfloor n/2 \rfloor$. So, the number of comparisons required is one less than the total number of elements in the set.

Step -3 : In order to find the minimum from SET B using linear searching technique, one needs to perform, $\lfloor n/2 \rfloor -1$ comparisons. This is because, the number of elements in the set is $\lfloor n/2 \rfloor$ So, the number of comparisons required is one less than the total number of elements in the set.

Hence, total number of comparisions is $\lfloor n/2 \rfloor + \lfloor n/2 \rfloor -1 + \lfloor n/2 \rfloor -1$
$= 3 \lfloor n/2 \rfloor -2$
$= \lfloor 3n/2 \rfloor -2$

Hence, proved.

Problem 5:
Page 107 Problem #4 :

Page 107 Problem #6 :

Page 107 Problem #10 :