Name : Fadi Alahmad Alomar

ID : 1200180049

Assignment 1: Recurrences

Question 1:

to know the asymptotic relation between $f(n)$ and $g(n)$ we need to find $\lim_{n\to\infty}\frac{f(n)}{g(n)}$ if it is equal to ∞ then $f(n)\in\Omega(g(n))$ if it is equal to 0 then $f(n)\in\mathrm{O}(g(n))$ if it is equal to a number then $f(n)\in\Theta(g(n))$

- $f(n) = 100n + \log n$, $g(n) = n + (\log n)^2$

$\lim_{n\to\infty}\frac{100n+\log n}{n+(\log n)^2} = \lim_{n\to\infty}\frac{100+\frac{1}{n}}{1+\frac{2\log n}{n}} = \frac{100}{1} = 100$

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

- $f(n) = \log n$, $g(n) = \log n^2$

$\lim_{n\to\infty}\frac{\log n}{\log n^2} = \lim_{n\to\infty}\frac{\log n}{2\log n} = \lim_{n\to\infty}\frac{1}{2} = \frac{1}{2}$

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

- $f(n) = \frac{n^2}{\log n}$, $g(n) = n(\log n)^2$

$\lim_{n\to\infty}\frac{\frac{n^2}{\log n}}{n(\log n)^2} = \lim_{n\to\infty}(\frac{n^2}{\log n} * \frac{1}{n(\log n)^2}) = \lim_{n\to\infty}\frac{n}{(\log n)^3} = \lim_{n\to\infty}\frac{1}{\frac{3(\log n)^2}{n}}$

$\lim_{n\to\infty}\frac{n}{3(\log n)^2} = \lim_{n\to\infty}\frac{1}{\frac{6\log n}{n}} = \lim_{n\to\infty}\frac{n}{6\log n} = \lim_{n\to\infty}\frac{1}{\frac{6}{n}} = \lim_{n\to\infty}\frac{n}{6} = \infty$

$\therefore f(n)\in\Omega(g(n))$

- $f(n) = (\log n)^{\log n}$, $g(n) = \frac{n}{\log n}$

put $\log n = m \therefore f(n) = m^m$, $g(n) = \frac{2^m}{m}$
$\lim_{m\to\infty}\frac{m^m * m}{n^m} = \lim_{m\to\infty} m(\frac{m}{2})^m = \infty$

$\therefore f(n)\in\Omega(g(n))$

- $f(n) = \sqrt{n}$, $g(n) = (\log n)^5$

$\lim_{n\to\infty} \frac{\sqrt{n}}{(\log n)^5} = \lim_{n\to\infty} (\frac{1}{2\sqrt{n}} * \frac{n}{5(\log n)^4}) = \lim_{n\to\infty} \frac{\sqrt{n}}{10(\log n)^4} = \lim_{n\to\infty} \frac{\sqrt{n}}{80(\log n)^3} =$
$\lim_{n\to\infty} \frac{\sqrt{n}}{480(\log n)^2} = \lim_{n\to\infty} \frac{\sqrt{n}}{1920 \log n} = \lim_{n\to\infty} (\frac{1}{2\sqrt{n}} * \frac{n}{1920}) = \lim_{n\to\infty} \frac{\sqrt{n}}{3840} = \infty$

$\therefore f(n) \in \Omega(g(n))$

- $f(n) = n2^n$, $g(n) = 3^n$

prove by induction that $f(n) \in \mathrm{O}(g(n))$
we need to prove that $f(n) \leq g(n)$
base step:
$n \geq 2$
$2 * 2^2 = 8$
$3^2 = 9$
$8 < 9$
induction hypothesis $f(n) \leq g(n)$, $n2^n \leq 3^n$
R.T.P. $f(n+1) \leq g(n+1)$

1. in order to reach $(n+1)2^{n+1}$ we need to multiply $n2^n$ by $\frac{2(n+1)}{n}$ and when $n \geq 2$ this term is smaller than 3
2. in order to get 3^{n+1} we need to multiply 3^n by 3
   from 1 and 2 if we multiply the left side of $n2^n \leq 3^n$ by $\frac{2(n+1)}{n}$ and the left side by 3 the equality should not change
   $\because f(n) \leq g(n) \therefore f(n+1) \leq g(n+1)$
   which proves $f(n) \in \mathrm{O}(g(n))$

- $f(n) = 2^{\sqrt{\log n}}$, $g(n) = \sqrt{n}$

$g(n) = (n)^{\frac{1}{2}} = (2^{\log n})^{\frac{1}{2}} = 2^{\frac{1}{2}\log n}$
$\lim_{n\to\infty} \frac{2^{\sqrt{\log n}}}{2^{\frac{1}{2}\log n}} = \lim_{n\to\infty} 2^{\sqrt{\log n} - \frac{1}{2}\log n}$
$\frac{1}{2}\log n$ will grow faster than $\sqrt{\log n}$
$\therefore \lim_{n\to\infty} 2^{\sqrt{\log n} - \frac{1}{2}\log n} = 0$

$\therefore f(n) \in O(g(n))$

Question 2:

- in term of **unit cost**:
  the loop is done $\log m = \log 2^n = n$ times and each loop we do $\mathrm{O}(1)$ work thus this algorithm

is $\mathrm{O}(n)$

- in term of **bit cost**:
  the loop is done $\log m$ times which is equal to $\log 2^n = n$
  and in each loop, we do a multiplication which takes $\mathrm{O}(l^2)$ where l is the number of bits, in each loop the value of y is 2^{2^i} where i is the loop number for us to find it is bit length we take the $\log_2 2^{2^i} = 2^i$
  $\therefore$ in each loop we do $O((2^i)^2) = O(2^{2i})$
  $\therefore$ the total work done is the sum of total work accross all levels = $\sum_{i=0}^{n} 2^{2i} = 2^2 * \frac{2^{2n}-1}{2^2-1} = O(2^{2n}) = O(4^2)$
  $\therefore$ the algorithm is $\mathrm{O}(4^n)$

Question 3:

we do BFS and put some indicators which alternates between two values that a node is visited and in each level search we enter we check if we have already visited the node that we are adding to the queue, we see if it is indicator is the opposite of the one we are currently in if it is the same then the graph is not a bipartite. if we visit all the nodes and no two adjacent nodes have the same indicator then the graph is bipartite.

in the graph structure each node has an array that include it is neighbours

```
FUNCTION Bipartite(node):
  myHash <- Hash()
  myQueue <- Queue()
  c <- -1
  myHash[node] <-c
  while myQueue.size != 0 do:
      current <- myQueue.pop()
      for i=0,i<current.neighbours.size do:
          if myHash[current.neighbours[i]] do:
              if myHash[current.neighbours[i]] != c*-1 do:
                  return False
              else do:
                  continue
          else do:
              myHash[current.neighbours[i]] = c*-1
          end
          myQueue.push(current.neighbours[i])
      end
      c <- c*-1
  end
  return True
```

the algorithm is $\mathrm{O}(n)$ where n is the number of nodes in the graph.

Question 4:

Expression	**Dominant term(x)**	$\mathrm{O}(...)$
$5 + 0.001n^3 + 0.025n$	$0.001n^3$	$\mathrm{O}(n^3)$
$500n + 100n^{1.5} + 50n\log_{10} n$	$100n^{1.5}$	$\mathrm{O}(n^{1.5})$
$0.3n + 5n^{1.5} + 2.5n^{1.75}$	$2.5n^{1.75}$	$\mathrm{O}(n^{1.75})$
$n^2 \log_2 n + n(\log_2 n)^2$	$n^2 \log_2 n$	$\mathrm{O}(n^2 \log n)$
$n\log_3 n + n\log_2 n$	$n\log_2 n$	$\mathrm{O}(n\log n)$
$3\log_8 n + \log_2 \log_2 \log_2 n$	$3\log_8 n$	$\mathrm{O}(\log n)$
$100n + 0.01n^2$	$0.01n^2$	$\mathrm{O}(n^2)$
$0.01n + 100n^2$	$100n^2$	$\mathrm{O}(n^2)$
$2n + n^{0.5} + 0.5n^{1.25}$	$0.5n^{1.25}$	$\mathrm{O}(n^{1.25})$
$0.01n\log_2 n + n(\log_2 n)^2$	$n(\log_2 n)^2$	$\mathrm{O}(n(\log n)^2)$
$100n\log_2 n + n^3 + 100n$	n^3	$\mathrm{O}(n^3)$
$0.003\log_4 n + \log_2 \log_2 n$	$0.003\log_4 n$	$\mathrm{O}(\log n)$

Question 5:

- Rule of sums: $O(f+g) = O(f) + O(g)$
  False
  Correct formula: $O(f) + O(g) = O(max(f,g))$
- Rule of products: $O(f*g) = O(f) * O(g)$
  True
- Transitivity: if $g = O(f)$ and $h = O(f)$ then $g = O(h)$
  False
  Correct formula: if $g = O(f)$ and $f = O(h)$ then $g = O(h)$
- $5n + 8n^2 + 100n^3 = O(n^4)$
  True

- $5n + 8n^2 + 100n^3 = O(n^2 \log n)$
  False
  Correct formula: $5n + 8n^2 + 100n^3 = O(n^3)$

Question 6:

- $f_1(n) = n^{0.999999} \log n$, $f_2(n) = 10000000n$, $f_3(n) = 1.000001n$, $f_4(n) = n^2$
  using the products rule $O(n^{0.999999} \log n) = O(n^{0.999999}) * O(\log(n))$
  $\because \log(n) = O(n^c)$ where $c > 0$
  $\therefore O(\log(n)) = O(n^{0.000001})$
  $\therefore O(n^{0.999999} \log n) = O(n^{0.999999}) * O(n^{0.000001}) = O(n^{0.999999} * n^{0.000001}) = O(n)$
  $O(f_4) > O(f_1) = O(f_2) = O(f_3)$

- $f_1(n) = 2^{2^{1000000}}$, $f_2(n) = 2^{100000n}$, $f_3(n) = \binom{n}{2} = \frac{n^2-n}{2}$, $f_4(n) = n\sqrt{n}$
  $O(f_2) > O(f_3) > O(f_4) > O(f_1)$

- $f_1(n) = n^{\sqrt{n}}$, $f_2(n) = 2^n$, $f_3(n) = n^{10} * 2^{n/2} = 2^{\log n^{10}} * 2^{\frac{n}{2}} = 2^{\frac{n}{2} + \log n^{10}}$, $f_4(n) = \sum_{i=1}^{n}(i+1) = \sum_{i=1}^{n} i + \sum_{i=1}^{n} 1 = \frac{n(n+1)}{2} + n = \frac{n^2}{2} + n + \frac{1}{2}$
  $O(f_2) > O(f_3) > O(f_1) > O(f_4)$