q3_answers

Wednesday, August 31, 2022 10:58 PM

Part (a)

$\Theta(\log(\log(n)))$

```
void f1(int n)
{
  int i=2;          ← O(1)
  while (i<n){      ← O(1)
    i=i*i;
  }
}
```

The while loop will run for as long as i is less than n. Every iteration, i is multiplied w/ itself and i's initial value is 2. For each iteration, k at the top of the while loop, $i = 2^{2^k}$. To find the k value where $i > n$, I set up an equation and solved for k.

$2^k > \log_2(n)$

$k > \log_2(\log_2(n))$

$\Theta(\log(\log(n)))$

$2^1 * 2^1 = 2^2$ $\quad 2^{2^k} > n$

$2^2 * 2^2 = 2^4$ $\quad 2^k \log_2(2) > \log_2(n)$

$2^3 * 2^3 = 2^6$

$n=36$ $\sqrt{n}=6$

$n=25$ $\sqrt{n}=5$

$n=100$ $\sqrt{n}=10$

$n=64$ $\sqrt{n}=8$

Part (b)

```
void f2(int n)
{
  for (int i=1; i<=n; i++){
    if ((i % (int)sqrt(n)) == 0){
      for (int k=0; k<pow(i,3); k++){   /* O(1) */   } O(i^3)
    }
  }
}
```

The innermost for loop only runs on the condition that $i \% \sqrt{n} = 0$. Otherwise, there is no work done in the function so to evaluate runtime, we need to determine how many times the inner for loop runs and what the runtime of the inner for loop is.

The inner loops runs $\sqrt{n}$ times. The runtime of the inner for loop is i^3 where in each iteration of the inner for loop, $i = k \cdot \sqrt{n}$ where k is the iteration in which the inner for loop is ran (based on the condition of the if statement).

I completed & simplified the sum to find the runtime...

$\sum_{k=1}^{\sqrt{n}} \Theta(k \cdot \sqrt{n})^3$

$= \sum_{k=1}^{\sqrt{n}} \Theta(k^3 \cdot \sqrt{n}^3)$ distribute

$= \sqrt{n}^3 \sum_{k=1}^{\sqrt{n}} \Theta(k^3)$ arithmetic series

$= \sqrt{n}^3 \Theta(\sqrt{n})^4$

$= \Theta(n^{7/2})$

$= n^{\frac{3}{2}} \Theta(n^2)$

$= n^{7/2}$

$O(n^{7/2})$

$O(i^3)$ if $i \% \sqrt{n} = 0$

$\Omega(n)$

$\Theta(n) + \sqrt{n}\,\Theta(i^3)$

~~$\Theta(n \cdot \sqrt{n}) + \sqrt{n}\Theta(i^3)$~~

$\sum_{i=0}^{n} \left(\sum_{j=0}^{\sqrt{n}} \sum_{k=0}^{i^3} O(1) \right)$ → $O(i^3)$

$n=16$

$i=1$

$\vdots$

✓ $i=4$

$\vdots$

✓ $i=8$

$\vdots$

✓ $i=12$

$\vdots$

✓ $i=16$

4^3, 8^3, 12^3, 16^3

$(1 \cdot \sqrt{n})^3 + \cdots + (k \cdot \sqrt{n})^3$

$4^3 + 8^3 + 12^3 + 16^3$

$(1 \cdot \sqrt{n})^3 + (2 \cdot \sqrt{n})^3 + (3 \cdot \sqrt{n})^3 + (4 \cdot \sqrt{n})^3$

Part (c)

```
for (int i=1; i<=n; i++){
  for (int k=1; k<=n; k++){
    if (A[k]==i){                                   O(n)
      for (int m=1; m<=n; m=m+m){ // O(1)           Θ(inner loop) ⇒ log2(n)
        //A[i] is const
      }
    }
  }
}
```

How often does $A[k]==i$? Well when it does

a. $m=1$
b. $m=2$
c. $m=4$
d. $m=8$
e. $m=16$
f. $m=32$

20 steps @ $m>=n$

$2^k >= n$

$k = \log_2(n)$

"A" has only $\#$ → $\sum_{k=1}^{n} \log_2(n) + \sum_{i=2}^{n} \sum_{k=1}^{n} O(1) + \sum_{i=2}^{n} n$

$n\log(n) + n(n-1)$

$n\log(n) + n^2 - n$ → $O(n^2)$

Explanation: The worst case scenario would be where A is a matrix of all the same values. In this case, the if statement will only result in true for a single iteration of the outermost for loop. When the if statement is true, its for loop will run for $O(\log_2(n))$ because the value of m increases exponentially by powers of 2. After this single iteration where the if statement is triggered, the outer loop will run for $n-1$ times and the inner loop will run for n times with a run time of $O(1)$. I reduced the arithmetic from there and got a total runtime of

$O(n^2)$

Part (d)

$\sum_{i=0}^{n-1}$ size = 10;

```
for (int i=0; i<n; i++){
  if (i==size){
    int newsize = 3*size/2;
    int *b = new int[newsize];
    for (int j=0; j<size; j++) b[j] = a[j];   ← O(i)
    delete [] a; // deallocate a
    a = b; // a points to
    size = newsize;
  }
  a[i] = i*i;   ← O(1)
}
```

When if statement isn't triggered, only $O(1)$ runtime

iteration	1	2	3	4	5	6	7	8	9	10	11	12
size	10	15	22	33	50	75	113	170	256	384	576	
n	20	30	40	50	60	70	80	90–	100			
trig #	2	3	4	5	5	5	6	6				

120–128	170–256
7	8

if statement is triggered logarithmically

Values are growing like $10 * \frac{3^k}{2^k}$

$10 * \frac{3^k}{2^k} \cdot \text{size}^i$

$\frac{3}{2}^k = i/10$

$k = \log_{\frac{3}{2}}(i/10)$

- if statement is triggered logarithmically as shown in my work highlighted on the right
- loop accumulates $O(1)$ ea time for a total of n times

$\sum_{i=0}^{n} O(1) + \sum_{k=0}^{\log(i)} \left(\sum_{j=0}^{i} \Theta(1) \right)$

$\Theta(n) + \sum_{k=0}^{\log(i)} \Theta(i)$

$\Theta(n) + \Theta(\log(i)\,i)$

this doesn't trigger every iteration AND when it does, it's smaller than or equal to n

$\Theta(n)$ is final result