Design and Analysis of Computer Algorithms
CS 6363.005: Homework #1

Due on Monday September 12, 2016 at 11:59pm

Professor Benjamin Raichel

Hanlin He **(hxh160630)**

hanlin.he@utdallas.edu

Contents

Problem 1 Running Time Analysis

Part (a) BackForwardAlg(n)

$\Theta\left(\log_2(n)\right)$

Part (b) RecursiveAlg(n)

$\Theta(n^2)$

Part (c) Description of RecursiveAlg(n)

The *RecursiveAlg* was designed to find duplicate item in the array $A[1...n]$, by dividing the array in two part, comparing every item from each half, and then keeping doing the operation recursively.

Problem 2 Induction on Binary Trees

Part (a) Induction on *gbt*

Proof: Let $nodes(n)$ be the nodes number of a good binary tree *gbt*, n is the leaves number. We are required to prove $\forall n > 0$, $nodes(n) = 2n - 1$.

Base Case: $n = 1$: The only node in the tree is the leaf itself. The tree has $2n - 1 = 1$ node. Hence the claim holds true for $n = 1$.

Induction step: Let $k > 1$ be an arbitrary natural number.
Let us assume the induction hypothesis: for every *gbt* with i leaves, $0 < i \leq k$, assume $nodes(i) = 2i - 1$. We will prove $nodes(i + 1) = 2(i + 1) - 1$.
Since the tree is a *gbt*, every non-leaf node point two distinct *gbt*'s. If we want to add one leaves to the tree, we need to add two leaves to a leaf node, making the original leaf node a non-leaf node. Hence two nodes are added to the *gbt*, which mean

$$nodes(k + 1) = nodes(k) + 2 = 2k - 1 + 2 = 2(k + 1) - 1 \tag{2.1}$$

Thus establishes the claim for $k + 1$.

Conclusion: By the principle of mathematical induction, the claim holds for all $n \in \mathbb{N}$. □

Part (b) Discussion on *bbt*

It does not hold that a *bbt* with $n > 0$ leaves, has $2n - 1$ nodes in total.

Consider removing one leaf from a *gbt* only. The leaf's parent node now has one child point to a *bbt*, which is Null (i.e. the empty tree). Now there are $n-1$ leaves with $2n-2$ nodes, which indicates $nodes(n-1) = 2(n-1)$ for a *bbt*.
Thus, the upper bound possible is $2n$.

Problem 3 Asymptotic Bounds

The Asymptotic Bounds are as follow:

(a)

$$\sum_{i=1}^{n} \frac{n}{i} = \Theta\left(n \log(n)\right) \tag{3.1}$$

(b)

$$\sum_{i=1}^{n} i^3 = \Theta\left(\frac{n^2(n+1)^2}{4}\right) = \Theta\left(n^4\right) \tag{3.2}$$

(c)

$$\begin{aligned} \sum_{i=1}^{n} \log(\frac{n}{i}) &= \sum_{i=1}^{n} (\log(n) - \log(i)) \\ &= \sum_{i=1}^{n} (\log(n)) - \sum_{i=1}^{n} (\log(i)) \\ &= n \log(n) - \log n! \\ &\approx n \log n - (n \log n - n) \\ &= \Theta(n) \end{aligned} \tag{3.3}$$

(d)

$$\frac{2}{n} + \sqrt{n} + \log^3 n = \Theta\left(n^{\frac{1}{2}}\right) \tag{3.4}$$

(e)

$$Bits(n) = \log_2(10^n) = n \log_2(10) = \Theta(n) \tag{3.5}$$

(f)

$$P(100) = \Theta(1) \tag{3.6}$$

Problem 4 Ordering functions

- $n^{\frac{7}{(2n)}}$
- $\cos n + 2$
- $\lg^* (n/8)$, $\lg^* \left(2^{2^{2^n}}\right)$
- $(\lg^* n)^{\lg n}$
- $1 + \lg \lg \lg n$
- $12 + \lfloor \lg \lg(n) \rfloor$
- $\sqrt{\lg n}$
- $\lg n$
- $\lg^{201} n$
- $n^{3/(2 \lg n)}$
- $n^{1/ \lg \lg n}$
- $n^{\frac{1}{125}}$
- n, $2^{\lg n}$
- $n(\lg n)^4$
- $\sqrt{n}^e$
- $\sum_{i=1}^{n} i$
- $n^{2.1}$
- $n^{4.5} - (n-1)^{4.5}$
- $\sum_{i=1}^{n} i^2$
- n^5
- $n^{\lg \lg n}$
- $(\lg(2+n))^{\lg n}$
- $\left(1 + \frac{1}{154}\right)^{15n}$

Problem 5 UTD Parking

The recurrence is described in pseudo-code as followed:

```
procedure P(n)
    if n = 0 then                          ▷ If there is no space left.
        return 1
    else if n = 1 then                     ▷ If there is only 1 space left.
        return 1
    else          ▷ If there are more than 1 space, then there are two ways to park.
        return P(n − 1) + P(n − 2)
    end if
end procedure
```

Algorithm 1: Count Number of Distinct Strings

Problem 6 Bounding Recurrences

Part (a) $T(n) = 2T(n/2) + n^4$

$T(n) = 2T(n/2) + n^4$ indicates that at the i recursive level, the total operations are:

$$f_i(n) = 2^i \times \left(\frac{n}{2^i}\right)^4 = \frac{n^4}{2^{3i}} \tag{6.1}$$

The depth of the recursion tree is $\lg n$, hence, the total running time is:

$$T(n) = \sum_{i=0}^{\lg n} f_i(n) = \sum_{i=0}^{\lg n} \frac{n^4}{2^{3i}} \tag{6.2}$$

Now consider the ratio of successive level sums:

$$r = \frac{f_{i+1}(n)}{f_i(n)} = \frac{\frac{n^4}{2^{3(i+1)}}}{\frac{n^4}{2^{3i}}} = \frac{1}{8} < 1 \tag{6.3}$$

Which means the total running time $T(n)$ is mainly decided by the $T(root)$, i.e.

$$T(n) = \Theta(n^4) \tag{6.4}$$

Problem 6 continued on next page...

Part (b) $T(n) = 16T(n/4) + n^2$

$T(n) = 16T(n/4) + n^2$ indicates that at the i recursive level, the total operations are:

$$f_i(n) = 16^i * \left(\frac{n}{4^i}\right)^2 = n^2 \tag{6.5}$$

which is not related to the level i. As the depth of the recursion tree is $\log_4 n$, hence, the total running time is:

$$T(n) = \sum_{i=0}^{\log_4 n} f_i(n) = \sum_{i=0}^{\log_4 n} n^2 = n^2 \log_4 n = \Theta(n^2 \log n) \tag{6.6}$$

Part (c) $T(n) = 2T(n/3) + T(n/4) + n$

Assume $T(n) = 2T(n/3) + T(n/4) + n$ indicates that at the i recursive level, the total operations are $f_i(n)$. We can write the $f_i(n)$ as:

$$\begin{aligned}
f_0(n) &= n, \\
f_1(n) &= \left(\frac{2}{3} + \frac{1}{4}\right) n = \left(\frac{11}{12}\right) n, \\
f_2(n) &= \left(\frac{4}{3^2} + \frac{4}{3 \times 4} + \frac{1}{4^2}\right) n = \left(\frac{121}{144}\right) n = \left(\frac{11}{12}\right)^2 n, \\
f_3(n) &= \left(\frac{8}{3^3} + \frac{12}{3^2 \times 4} + \frac{6}{3 \times 4^2} + \frac{1}{4^3}\right) n = \left(\frac{1331}{1728}\right) n = \left(\frac{11}{12}\right)^3 n, \\
&\dots
\end{aligned} \tag{6.7}$$

According the pattern of $f_i(n)$ above, it is easy to conclude and prove by induction that the general form of $f_i(n)$ is:

$$f_i(n) = \left(\frac{11}{12}\right)^i n \tag{6.8}$$

Now consider the ratio of successive level sums:

$$r = \frac{f_{i+1}(n)}{f_i(n)} = \frac{\left(\frac{11}{12}\right)^{i+1} n}{\left(\frac{11}{12}\right)^i n} = \frac{11}{12} < 1 \tag{6.9}$$

Which means that the total running time $T(n)$ is mainly decided by the $T(root)$, i.e.

$$T(n) = \Theta(n) \tag{6.10}$$

Problem 6 continued on next page...

Part (d) $T(n) = T(n-1) + \sqrt{n}$

$T(n) = T(n-1) + \sqrt{n}$ indicates that at the i recursive level, the total operations are:

$$f_i(n) = \sqrt{n-i} \tag{6.11}$$

The depth of the recursion tree is n, hence, the total running time is:

$$T(n) = \sum_{i=0}^{n} f_i(n) = \sum_{i=0}^{n} \sqrt{n-i} = \sum_{i=1}^{n} \sqrt{n}$$

Now consider the ratio of successive level sums:

$$r = \frac{f_{i+1}(n)}{f_i(n)} = \frac{\sqrt{n-i+1}}{\sqrt{n-i}} < 1 \tag{6.12}$$

Which means that the total running time $T(n)$ is mainly decided by the $T(root)$, i.e.

$$T(n) = \Theta(\sqrt{n}) \tag{6.13}$$

Part (e) $T(n) = 3T(n/2) + 5n$

$T(n) = 3T(n/2) + 5n$ indicates that at the i recursive level, the total operations are:

$$f_i(n) = 3^i \times \left(\frac{5n}{2^i}\right) = \left(\frac{3}{2}\right)^i 5n \tag{6.14}$$

The depth of the recursion tree is $\lg n$, hence, the total running time is:

$$T(n) = \sum_{i=0}^{\lg n} \left(\frac{3}{2}\right)^i 5n = 5n \sum_{i=0}^{\lg n} \left(\frac{3}{2}\right)^i \tag{6.15}$$

Now consider the ratio of successive level sums:

$$r = \frac{f_{i+1}(n)}{f_i(n)} = \frac{\left(\frac{3}{2}\right)^{i+1} 5n}{\left(\frac{3}{2}\right)^i 5n} = \frac{3}{2} > 1 \tag{6.16}$$

Which means the total running time $T(n)$ is mainly decided by the $T(leaf)$, the last recursion's running time i.e.

$$T(n) = 5 \times 3^{\lg n} = \Theta(n^{\lg 3}) \tag{6.17}$$

Problem 6 continued on next page. . .

Part (f) $T(n) = T(\sqrt{n}) + 7$

$T(n) = T(\sqrt{n}) + 7$ indicates that at the i recursive level, the total operations are:

$$f_i(n) = 7 \tag{6.18}$$

The depth of the recursion tree is n, hence, the total running time is:

$$T(n) = \sum_{i=0}^{\sqrt{n}} f_i(n) = \sum_{i=0}^{\sqrt{n}} 7 \tag{6.19}$$

In this case, the ratio of successive level sums is apparently $r = 1$, which means that the total running time $T(n)$ is mainly decided by the depth of the recursion tree.
Assume the base case is $n = 1$, $\sqrt[2i]{n}$, in which i is a natural number representing the *ith* recursion, will not reach 1, though $\lim_{i=0}^{\infty} \sqrt[2i]{n} = 1$. Thus, the recursion will not reach base case, i.e.

$$T(n) = \Theta(\infty) \tag{6.20}$$