Design and Analysis of Algorithms
CS575, Spring 2023

Theory Assignment 2.1

Due on 3/2/2023 (Thursday)

1. (16 points) Use the iteration method or recursion tree method to solve the following recurrence equation.

   a) (8 points)

$$T(n) = \begin{cases} T(n-1) + n & if\ (n > 1) \\ 1 & if\ (n = 1) \end{cases}$$

   b) (8 points) You can assume $n^{1/2^k} = 2$ for some integers *n* and *k*.

$$T(n) = \begin{cases} 0 & if\ n = 2 \\ T(\sqrt{n}) + 1 & if\ n > 2 \end{cases}$$

$$T(n) = \begin{cases} T(n-1) + n & \text{if } (n > 1) \\ 1 & \text{if } (n = 1) \end{cases}$$

Lets solve this using the iteration method and by expanding the relation for small 'n's', look for a pattern.

$n = 1$, $T(1) = 1$
$n = 2$, $T(2) = T(1) + 2 = 3$
$n = 3$, $T(3) = T(2) + 3 = 6$
$n = 4$, $T(4) = T(3) + 4 = 10$

It be seen that the sum of first 'n' natural no-s is involved.

The sum of first 'n' natural nos $= n(n+1)/2$

$$\begin{aligned} T(n) &= T(n-1) + n \\ &= T(n-2) + (n-1) + n \\ &= T(n-3) + (n-2) + (n-1) + n \\ &\vdots \\ &= T(1) + 2 + 3 + \dots + n \end{aligned}$$

Therefore, the solution of the recurrence relation is $T(n) = \Theta(n^2)$.

- You can assume $n^{1/2^k} = 2$ for some integers n & k

$$T(n) = \begin{cases} 0 & \text{if } n = 2 \\ T(\sqrt{n}) + 1 & \text{if } n > 2 \end{cases}$$

Recursion Tree method can be used to solve the recurrence relation.
At each level recursion, we take the square root of the input size n.

Let 'c' be the cost of each level of recursion.
$T(n)$ is the cost at the root.
$T(\sqrt{n})$ is the cost at the next level and so on.
~~Until~~ Until we reach the leaves at $T(2)$ with cost 0

$T(n) = c + T(\sqrt{n})$

$T(\sqrt{n}) = c + T(2)$

$T(2) = 0$
Substituting $T(2) = 0$ & $T(\sqrt{n}) = c + T(2)$ in the first equation

$\therefore T(n) = c + c + T(2\sqrt{n})$

Continuing this process we get

$T(n) = \log \log n * c + T(2)$

Since $T(2) = 0$
$T(n) = \Theta(\log \log n)$

Here's what the tree would look like.

$$T(n)$$

$$T(\sqrt{n}) + 1$$

$$T(\sqrt{\sqrt{n}}) + 1 \qquad T(\sqrt{\sqrt{n}}) + 1$$

$$T(\sqrt{\sqrt{\sqrt{n}}}) + 1 \quad T(\sqrt{\sqrt{\sqrt{n}}}) + 1 \quad T(\sqrt{\sqrt{\sqrt{n}}}) + 1 \quad T(\sqrt{\sqrt{\sqrt{n}}}) + 1$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$T(2) + 1 \quad T(2) + 1 \quad T(2) + 1 \quad T(2) + 1$$

2. (6 points) Use Master method to solve $T(n) = 4T(n/2) + n^2$ and T(1)=1.

2) $T(n) = 4T(n/2) + n^2$ & $T(1) = 1$.

$a = 4$, $b = 2$, $c = 1$ & $k = 2$

Calculating $b^k = 2^2$

$b^k = 4$

As $a = b^k$, 2nd case of Master Theorem comes into picture.

$T(n) = \theta(n^k \lg n)$

as $k = 2$

$\cancel{T(n) = n^2 \lg}$

$T(n) = \theta(n^2 \lg n)$

3. (10 points) Professor Caear wishes to develop a matrix-multiplication algorithm that is asymptotically faster than Strassen's Algorithm. His algorithm will use divide-and conquer method, dividing each matrix into pieces of size n/4× n/4, and the divide and combine steps together will take $\theta(n^2)$ time. He needs to determine how many subproblems his algorithm has to create in order to beat Strassen's algorithm. If his algorithm creates a subproblems, then the recurrence for the running time T(n) becomes T(n) = aT(n/4)+ $\theta(n^2)$. What is the largest integer value of a for which Professor Caesar's algorithm would be asymptotically faster than Strassen's algorithm?

3) The goal of Prof. ~~Caer~~ Caesar is to develop a matrix - multiplication algorithm, asymptotically faster than Strassen's algorithm.

The divide - conquer technique will be used by the algorithm, dividing each matrix into pieces of size $n/4 \times n/4$

These divide-and conquere steps will be taking $\Theta(n^2)$ time

For beating Strassen's algorithm, he needs to determine how many subproblems his algorithm has to create. Suppose his algorithm creates 'a' subproblems, then the recurrence for the running time becomes

$$T(n) = a\,T(n/4) + \Theta(n^2)$$

Asymptotic running time of Strassen's Algorithm is

$$S(n) = \Theta(n^{\lg_2 7})$$

Using Master's Theorem for Caesar's algorithm.

$b = 4 \quad k = 2 \quad \therefore b^k = 4^2 = 16$

Now depending on 'a'

$\Theta(n^2)$ if $a < 16$ (beats Strassen's algo)

$\Theta(n^2 \log n)$ if $a = 16$ (also better than Strassen's)

$\Theta(n^{\log_4 a})$ if $a > 16$

For this case we need that

$$\frac{\log a}{\log 4} < \frac{\log 7}{\log 2}$$

$$\frac{\log a}{\log 7} < \frac{\log 4}{\log 2}$$

$$\frac{\log a}{\log 7} < \log_2 2^2$$

$$\frac{\log a}{\log 7} < 2$$

$$\log a < \log 7^2$$

$$a < 49$$

The largest integer for a = 48.