Course: CUS715, Instructor: Dr. Christoforos Christoforou

Problem Set 6 - Solving Recurrences - Iteration (unfolding) method.

Important: When answering the questions below, make sure you follow the proof-format covered in class. That is, CLEARLY identify each step in the proof, show all algebraic steps, and explain any non-trivial steps.

Problems to solve:

1. Solve the following recurrence using the **Iteration method**. Show all steps.

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2.T(\tfrac{n}{2}) + 1000n \;\; \forall n \geq 2 \end{aligned}$$

Answer:

k = $\frac{n}{2}$

T(k) = 2T($\frac{n}{2^2}$) + 1000($\frac{n}{2}$)

T(n) = 2[2T($\frac{n}{2^2}$) + 1000($\frac{n}{2}$)] + 1000n

= 2^2T($\frac{n}{2^2}$) + 2000($\frac{n}{2}$) + 1000n

k = $\frac{n}{2^2}$

T(k) = 2T($\frac{n}{2^3}$) + 1000($\frac{n}{2^2}$)

T(n) = 2^2[2T($\frac{n}{2^3}$) + 1000($\frac{n}{2^2}$)] + 2000($\frac{n}{2}$) + 1000n

= 2^3T($\frac{n}{2^3}$) + 4000($\frac{n}{2^2}$) + 2000($\frac{n}{2}$) + 1000n

k = $\frac{n}{2^3}$

T(k) = 2T($\frac{n}{2^4}$) + 1000($\frac{n}{2^3}$)

T(n) = $2^3[2\mathrm{T}(\frac{n}{2^4}) + 1000(\frac{n}{2^3})]$+ 4000($\frac{n}{2^2}$) + 2000($\frac{n}{2}$) + 1000n

= $2^4\mathrm{T}(\frac{n}{2^4}) + 8000(\frac{n}{2^3})$+ 4000($\frac{n}{2^2}$) + 2000($\frac{n}{2}$) + 1000n

The next iteration, if we follow the pattern thus far, should be this:

$2^5\mathrm{T}\left(\frac{n}{2^5}\right) + 16000\left(\frac{n}{2^4}\right) + 8000(\frac{n}{2^3})$+ 4000($\frac{n}{2^2}$) + 2000($\frac{n}{2}$) + 1000n

T(n) = $2^k\mathrm{T}\left(\frac{n}{2^k}\right) + (\mathrm{k} - 1)(1000\mathrm{n})$

$\frac{n}{2^k}$ = 1

$n = 2^k$

$k = \log_2 n$

T(n) = n * T(1) + $((\log_2 n) - 1)$(1000n)

= n + (1000n$\log_2 n$) – 1000n

= (1000n$\log_2 n$) – 999n

T(n) is O(n$\log_2 n$)

2. Solve the following recurrence using the **Iteration method**. Show all steps.

$$
\begin{aligned}
T(1) &= 1 \\
T(n) &= 7.T(\tfrac{n}{2}) + 18n^2 \quad \forall n \geq 2
\end{aligned}
$$

3. Calculate the running time of the following recursive functions. Solve any recurrences that arise using the **iteration method**.

Answer:

T(1) = 1, T(n) = T($\frac{n}{2}$) + n

k = $\frac{n}{2}$

T(k) = T($\frac{n}{2^2}$) + $\frac{n}{2}$

T(n) = T($\frac{n}{2^2}$) + $\frac{n}{2}$ + n

k = $\frac{n}{2^2}$

T(k) = T($\frac{n}{2^3}$) + $\frac{n}{2^2}$

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

Following the pattern thus far…

T(n) = T($\frac{n}{2^k}$) + n$\sum_{i=0}^{k-1}(\frac{1}{2})^i$

$\sum_{i=0}^{k-1}(\frac{1}{2})^i$ = $\frac{1}{1-\frac{1}{2}}$ = $\frac{1}{\frac{1}{2}}$ = 2

$T(n)$ = 1 + 2n

$T(n)$ is O(n)

```
   public int recursive(int n)
{      int sum=0;      for (int
i=1; i<= n; i++){}        sum=
sum +1;
```

```
    }      if (n>1)
{         return
recursive(n/2);
    }
else
{        return
1
    }
  }
```

Deliverable

If you have not already done so, create a GitHub repository that will host all your assignment for this course. In the repository should create a folder titled `cus715_problem_set_06`. Within that folder, add a `pdf` or `markdown` document with your answers to the problem set.