COMP2119 Introduction to Data Structures and Algorithms Assignment 1 - Recursion, Mathematical Induction and Algorithm Analysis

Cheng Ho Ming, Eric (3036216734) [Section 1C, 2024]

September 17, 2024

1 Asymptotic Bounds [20%]

$\Theta(\log n) : (d)\ \log n, (j)\ \log \frac{n}{\log n},$
$\Theta(\frac{n}{\log n}) : (h)\ \frac{n}{\log n},$
$\Theta(\frac{n}{\log \log n}) : (i)\ \frac{n}{\log \log n},$
$\Theta(n) : (g)\ \frac{n}{\log \pi}, (m) \sqrt{\sum_{i=1}^{n}(i+1)},$
$\Theta(n^{\log \pi}) : (e)\ \pi^{\log n}, (f)\ n^{\log \pi}, (k)\ \pi^{\log(2n)},$
$\Theta(n^{\log 2\pi}) : (l)\ n^{\log 2\pi},$
$\Theta(n^{\pi}) : (a)\ n^{\pi},$
$\Theta(\pi^{n}) : (b)\ \pi^{n},$
$\Theta(n^{n}) : (n)\ 1910n! + 316n^{n}, (c)\ n^{n}$

2 Recurrence Relations [20%]

(a)
$$\begin{aligned} T(n) &= T(n-1) + 3 \quad \text{for } n > 0 \\ &= T(n-2) + 3 + 3 \quad \text{for } n > 1 \\ &= \dots \\ \therefore T(n) &= T(0) + 3n \\ &= 3n \\ \therefore T(n) &= \Theta(n) \end{aligned}$$

(b) Assume that n is a power of 3, i.e. $n = 3^k$ for $k \in \mathbb{N}$, and $\log_3 n = k$,

$$\begin{aligned} \because T(n) &= 3T(\frac{n}{3}) + n \quad \text{for } n \neq 1 \\ &= 3(3T(\frac{n}{9}) + \frac{n}{3}) + n \quad \text{for } n >= 9 \\ &= 9T(\frac{n}{9}) + n + n \quad \text{for } n >= 9 \\ &= ... \\ \therefore T(n) &= 3^k T(\frac{n}{3^k}) + k * n \\ &= 3^k T(1) + k * n \\ &= 0 + k * n \\ &= kn \\ &= n * \log_3 n \\ \therefore T(n) &= \Theta(n \log n) \end{aligned}$$

(c) Assume that n is a power of 3, i.e. $n = 3^k$ for $k \in \mathbb{N}$, and $\log_3 n = k$,

$$\begin{aligned}
\because T(n) &= 4T(\frac{n}{3}) + 1 \quad \text{for } n >= 3\\
&= 4(4T(\frac{n}{9}) + 1) + 1 \quad \text{for } n >= 9\\
&= 16T(\frac{n}{9}) + 4^{k-1} + ... + 4^0 \quad \text{for } n >= 9\\
&= ...\\
\therefore T(n) &= 4^k T(\frac{n}{3^k}) + \frac{1 * (1 - 4^k)}{1 - 4}\\
&= 4^k T(1) + \frac{4^k - 1}{3}\\
&= 0 + \frac{4^{log_3 n} - 1}{3}\\
&= \frac{1}{3} * 4^{log_3 n} - \frac{1}{3}\\
&= \frac{1}{3} * 3^{(log_3 4)(log_3 n)} - \frac{1}{3}\\
&= \frac{1}{3} * n^{log_3 4} - \frac{1}{3}\\
\therefore T(n) &= \Theta(n^{log_3 4})
\end{aligned}$$

(d) Assume that n is a power of 2, i.e. $n = 2^k$ for $k \in \mathbb{N}$, and $\log_2 n = k$,

$$\begin{aligned}
\because T(n) &= nT(\frac{n}{2}) + n - 1 \quad \text{for } n >= 2\\
&= n * (\frac{n}{2} * T(\frac{n}{4}) + \frac{n}{2} - 1) + n - 1 \quad \text{for } n >= 4\\
&= \frac{n^2}{2} * T(\frac{n}{4}) + \frac{n^2}{2} - n + n - 1 \quad \text{for } n >= 4\\
&= \frac{n^2}{2} * T(\frac{n}{4}) + \frac{n^2}{2} - 1 \quad \text{for } n >= 4\\
&= \frac{n^2}{2} * (\frac{n}{4} * T(\frac{n}{8}) + \frac{n}{4} - 1) + \frac{n^2}{2} - 1 \quad \text{for } n >= 8\\
&= \frac{n^3}{2 * 4} * T(\frac{n}{8}) + \frac{n^3}{2 * 4} - \frac{n^2}{2} + \frac{n^2}{2} - 1 \quad \text{for } n >= 8\\
&= \frac{n^3}{2^1 * 2^2} * T(\frac{n}{2^1 * 2^2}) + \frac{n^3}{2^1 * 2^2} - 1 \quad \text{for } n >= 8\\
&= \dots\\
\therefore T(n) &= \frac{n^k}{2^0 * 2^1 * 2^2 * ... * 2^{k-1}} * T(1) + \frac{n^k}{2^0 * 2^1 * 2^2 * ... * 2^{k-1}} - 1\\
&= \frac{n^k}{2^{\frac{(k-1)*k}{2}}} * 2 - 1\\
&= \frac{n^k}{n^{\frac{k-1}{2}}} * 2 - 1\\
&= n^{\frac{2k}{2} - \frac{k-1}{2}} * 2 - 1\\
&= n^{\frac{k+1}{2}} * 2 - 1\\
&= 2n^{\frac{log_2 n + 1}{2}} - 1\\
\therefore T(n) &= \Theta(n^{\frac{log_2 n + 1}{2}})
\end{aligned}$$

3 Mathematical Induction [30%]

(a) Let $f(n)$ be the predicate "$1 * 2^1 + 2 * 2^2 + 3 * 2^3 + ... + n * 2^n = (n-1)2^{n+1} + 2$" for $\forall n \in \mathbb{Z}^+$.

$$\begin{aligned}
\text{For } n = 1, \text{L.H.S.} &= 1 * 2^1 \\
&= 2 \\
\text{R.H.S.} &= (1-1)2^{1+1} + 2 \\
&= 2
\end{aligned}$$

$\because$ L.H.S. = R.H.S.

$\therefore f(1)$ is true.

Inductive Step:
Assume that $f(n)$ is true when $n = k$, for some $k \in \mathbb{Z}^+$, i.e. $f(k) = 1*2^1 + 2*2^2 + 3*2^3 + ... + k*2^k = (k-1)2^{k+1} + 2$.

$$\begin{aligned}
\text{Consider the case } n = k+1,\ \text{L.H.S.} &= 1 * 2^1 + 2 * 2^2 + 3 * 2^3 + ... + k * 2^k + (k+1) * 2^{k+1} \\
&= (k-1)2^{k+1} + 2 + (k+1) * 2^{k+1} \quad \text{(by induction hypothesis)} \\
&= 2^{k+1} * (2k) + 2 \\
&= (k)2^{k+2} + 2 \\
&= (k+1-1)2^{k+1+1} + 2 \\
&= \text{R.H.S.}
\end{aligned}$$

$\because f(n)$ is true when $n = k+1$.

$\therefore$ By the principle of mathematical induction, $f(n)$ is true for all $n \in \mathbb{Z}^+$.

(b) Let $f(n)$ be the predicate " $\frac{1}{\sqrt{1}+\sqrt{2}} + \frac{1}{\sqrt{2}+\sqrt{3}} + \frac{1}{\sqrt{3}+\sqrt{4}} + ... + \frac{1}{\sqrt{n}+\sqrt{n+1}} = \sqrt{n+1} - 1$ for $\forall n \in \mathbb{Z}^+$ ".

$$\begin{aligned}
\text{For } n = 1, \text{L.H.S.} &= \frac{1}{\sqrt{1}+\sqrt{2}} \\
&= \frac{1}{\sqrt{1}+\sqrt{2}} * \frac{\sqrt{1}-\sqrt{2}}{\sqrt{1}-\sqrt{2}} \\
&= \frac{\sqrt{1}-\sqrt{2}}{1-2} \\
&= \sqrt{2} - 1 \\
\text{R.H.S.} &= \sqrt{1+1} - 1 \\
&= \sqrt{2} - 1
\end{aligned}$$

$\because$ L.H.S. = R.H.S.

$\therefore f(1)$ is true.

Inductive Step:
Assume that $f(n)$ is true when $n = k$, for some $k \in \mathbb{Z}^+$, i.e. $f(k) = \frac{1}{\sqrt{1}+\sqrt{2}} + \frac{1}{\sqrt{2}+\sqrt{3}} + \frac{1}{\sqrt{3}+\sqrt{4}} + ... + \frac{1}{\sqrt{k}+\sqrt{k+1}} = \sqrt{k+1} - 1$.

Consider the case $n = k+1$,

$$\begin{aligned}
\text{L.H.S.} &= \frac{1}{\sqrt{1}+\sqrt{2}} + \frac{1}{\sqrt{2}+\sqrt{3}} + \frac{1}{\sqrt{3}+\sqrt{4}} + ... + \frac{1}{\sqrt{k}+\sqrt{k+1}} + \frac{1}{\sqrt{k+1}+\sqrt{k+2}} \\
&= \sqrt{k+1} - 1 + \frac{1}{\sqrt{k+1}+\sqrt{k+2}} \quad \text{(by induction hypothesis)} \\
&= \sqrt{k+1} - 1 + \frac{1}{\sqrt{k+1}+\sqrt{k+2}} * \frac{\sqrt{k+1}-\sqrt{k+2}}{\sqrt{k+1}-\sqrt{k+2}} \\
&= \sqrt{k+1} - 1 + \frac{\sqrt{k+1}-\sqrt{k+2}}{1-2} \\
&= \sqrt{k+1} - 1 + \sqrt{k+2} - \sqrt{k+1} \\
&= \sqrt{k+2} - 1 \\
&= \sqrt{k+1+1} - 1 \\
&= \text{R.H.S.}
\end{aligned}$$

$\because f(n)$ is true when $n = k+1$.

$\therefore$ By the principle of mathematical induction, $f(n)$ is true for all $n \in \mathbb{Z}^+$.

4 Algorithm Design [30%]

The algorithm is:
For $n \in \mathbb{Z}^+$,

$$f(n) = 2^n = \begin{cases} 2, & \text{if } n = 1, \\ f(\frac{n}{2})^2, & \text{if } n \text{ is even,} \\ 2 * f(n-1), & \text{if } n \text{ is odd and } n > 1. \end{cases}$$

Correctness of the algorithm:

Base case:

For $n = 1, f(1) = 2^1 = 2,$ which is correct.

Inductive Step:

$$\text{For } n = 2k, \text{ where } k \in \mathbb{Z}^+$$
$$f(2k) = f(k)^2 = (2^k)^2 = 2^{2k} = 2^{2k}, \text{ which is correct.}$$

$$\begin{aligned} &\text{For } n = 2k+1, \text{ where } k \in \mathbb{Z}^+ \\ &f(2k+1) = 2 * f(2k+1-1) = 2 * f(2k) \\ &\qquad = 2 * f(k)^2 \\ &\qquad = 2 * 2^{2k} \\ &\qquad = 2^{2k+1}, \text{ which is correct.} \end{aligned}$$

$\because$ f(2k) is true if f(k) is true, and f(2k+1) is true if f(2k) is true.
$\therefore$ By the principle of mathematical induction, the algorithm is correct.

Running time of the algorithm:

$$\begin{aligned} f(n) &= f(\frac{n}{2})^2, \text{ if n is even} \\ &= f(\frac{n}{4})^{2^2}, \text{ if } \frac{n}{2} \text{ is even} \\ &= \dots \\ &= f(\frac{n}{2^k})^{2^{2^{\cdots}} \text{ <- k amount of "2"s}}, \text{ for } n = 2^k \text{ such that } k = \log_2 n \\ &= 2^{2^{2^{\cdots}}} \end{aligned}$$

$\because$ in this case, our algorithm needs to perform $k = \log_2 n$ squaring operations, and each squaring operation takes $O(1)$ time (as the typical algorithm for squaring a number: $2^n = 2 * 2^{n-1}$ for $n \in \mathbb{Z}^+$ and $n > 1$ takes $O(n)$ time).
$\therefore f(n) = O(\log n)$ if n is even.

$$\begin{aligned} \text{If } n \text{ is odd and } n > 1, f(n) &= 2 * f(n-1) \\ &= 2 * O(f(n-1)) \text{ where } n-1 \text{ is an even number,} \\ &= O(f(n-1)), \text{ which is given to be } O(\log n). \end{aligned}$$

$\therefore$ the running time of the algorithm is $O(\log n)$, which is faster than $O(n)$.