COMP2119 Introduction to Data Structures and Algorithms
Practice Problem Set 1 - Recursion, Mathematical Induction & Algorithm Analysis

Please do NOT submit the answer of this part.

1. For each of the following recurrence relations, solve the recurrence and state the Θ bound.

   (a) $T(n) = T(n-2) + n$; $T(0) = T(1) = 0$

   (b) $T(n) = T(\sqrt{n}) + 2$; $T(1) = T(2) = 0$ (You may assume that $n = 2^{2^k}$ where k is a power of 2.)

   (c) $T(n) = 5T(n/5) + n/5$; $T(1) = 0$ (You may assume that n is a power of 5.)

   (d) $T(n) = T(n-1) + \log_2 n$; $T(1) = 0$

   (e) $T(n) = T(\frac{2n}{5}) + T(\frac{3n}{5}) + n$; $T(n) = 0$ if $n \leq 1$

Solution:

(a) Case 1: n is even number and $n >= 0$:

$$\begin{aligned} T(n) &= T(n-2) + n \\ &= T(n-4) + (n-2) + n \\ &= T(n-6) + (n-4) + (n-2) + n \\ &= 0 + 2 + ... + (n-4) + (n-2) + n \\ &= \frac{n(n+2)}{4} \rightarrow \Theta(n^2) \end{aligned}$$

Case 2: n is odd number and $n >= 0$:

$$\begin{aligned} T(n) &= T(n-2) + n \\ &= T(n-4) + (n-2) + n \\ &= T(n-6) + (n-4) + (n-2) + n \\ &= 0 + 3 + ... + (n-4) + (n-2) + n \\ &= \frac{(n+1)^2}{4} - 1 \rightarrow \Theta(n^2) \end{aligned}$$

Therefore, the Θ bound is $\Theta(n^2)$.

(b)

$$\begin{aligned} T(n) &= T(\sqrt{n}) + 2 \\ &= T(n^{\frac{1}{2}}) + 2 \\ &= T(n^{\frac{1}{2^2}}) + 2 * 2 \\ &= T(n^{\frac{1}{2^3}}) + 2 * 3 \\ &= T(n^{\frac{1}{2^k}}) + 2 * k \quad \text{(Assume that } n = 2^{2^k}\text{)} \\ &= T(n^{\frac{1}{2^{\log_2 \log_2 n}}}) + 2 * (\log_2 \log_2 n) \\ &= T(2) + 2 * (\log_2 \log_2 n) \\ &= 0 + 2 * (\log_2 \log_2 n) \rightarrow \Theta(\log \log n) \end{aligned}$$

(c) $T(n) = 5T(\frac{n}{5}) + \frac{n}{5}$

$= 5[5T(\frac{n}{5^2}) + \frac{n}{5^2}] + \frac{n}{5}$

$= 5^2T(\frac{n}{5^2}) + \frac{n}{5} * 2$

$= 5^2[5T(\frac{n}{5^3}) + \frac{n}{5^3}] + \frac{n}{5} * 2$

$= 5^3T(\frac{n}{5^3}) + \frac{n}{5} * 3$

$= 5^kT(\frac{n}{5^k}) + \frac{n}{5} * k$ (Assume that $n = 5^k$)

$= 5^{\log_5 n}T(\frac{n}{5^{\log_5 n}}) + \frac{n}{5} * \log_5 n$

$= 5^{\log_5 n}T(1) + \frac{n}{5} * \log_5 n$

$= \frac{n}{5} * \log_5 n \to \Theta(n \log n)$

(d) $T(n) = T(n-1) + \log_2 n$

$= [T(n-2) + log_2(n-1)] + \log_2 n$

$= [T(n-3) + log_2(n-2)] + log_2(n-1) + \log_2 n$

$= T(n-3) + log_2(n-2) + log_2(n-1) + \log_2 n$

$= \log_2 1 + \log_2 2 + ... + \log_2 (n-2) + \log_2 (n-1) + \log_2 n$

$= \log_2 n! \to \Theta(n \log n)$

Proof of $\log_2 n! \to \Theta(n \log n)$:
Since it is obvious that $n^n >= n!$. We can see that $n! = 1 * 2 * .. * (n/2) * (n/2 + 1) * ... * n >= (n/2) * (n/2 + 1) * ... * n >= (n/2) * (n/2) * ... * (n/2) = (n/2)^{n/2}$. So $\log n^n >= \log n! >= \log((n/2)^{n/2})$, we have $n \log n >= \log n! >= (n/2) \log(n/2)$.

(e) $T(n) = T(\frac{2n}{5}) + T(\frac{3n}{5}) + n$ (level 1 in Figure 1)

$= T(n * (\frac{2}{5})^2) + T(\frac{6n}{25}) + T(\frac{6n}{25}) + T(n * (\frac{3}{5})^2) + n * 2$

$= T(\frac{n}{(\frac{5}{2})^2}) + T(\frac{n}{(\frac{5}{2}) * (\frac{5}{3})}) + T(\frac{n}{(\frac{5}{3}) * (\frac{5}{2})}) + T(\frac{n}{(\frac{5}{3})^2}) + n * 2$ (level 2 in Figure 1)

$= ...$

We can expand the recurrence equation as Figure 1. More details of the recursion tree can be found at chapter 2 and chapter 4 in "Introduction to Algorithms" by Leiserson, Cormen, Rivest and Stein, MIT press.

By observation, each $T(n)$ can split into two new $T(n)$s plus a single n according to the recurrence equation. In the last level of the recursion tree where all $T(n) = T(1) = 0$, we can see the equation is only related to the n part. It is easy to find that, at each level, the term that only contain n is n*(the current level).

Assume k is the total number of levels of the recursion tree, then Θ bound of $T(n)$ is $\Theta(nk)$. At the k-th level, the left-most $T(\frac{n}{(\frac{5}{2})^k})$ has the biggest denominator $(\frac{5}{2})^k$ than any other term relates to $T(n)$. So it will be reach to $T(1)$ earlier than other terms. Similarity, the right-most T is $T(\frac{n}{(\frac{5}{3})^k})$ has smallest denominator and will

be the last one to reach $T(1)$. The middle Ts have denominators between $(\frac{5}{3})^k$ and $(\frac{5}{2})^k$. Therefore, the final level k will be larger than the level when the left-most $T(\frac{n}{(\frac{5}{2})^k}) = T(1)$.

Similarly, k will be smaller than or equals to the level when the right-most $T(\frac{n}{(\frac{5}{3})^k}) = T(1)$. Then, we obtain $\lfloor \log_{\frac{5}{2}} n \rfloor \leq k \leq \lceil \log_{\frac{5}{3}} n \rceil$. The lower Θ bound is $\Theta(n * \log_{\frac{5}{2}} n)$ when $k = \lfloor \log_{\frac{5}{2}} n \rfloor$, and upper Θ bound is $\Theta(n * \log_{\frac{5}{3}} n)$ when $k = \lceil \log_{\frac{5}{3}} n \rceil$.

Therefore, the Θ bound is $\Theta(n \log n)$.

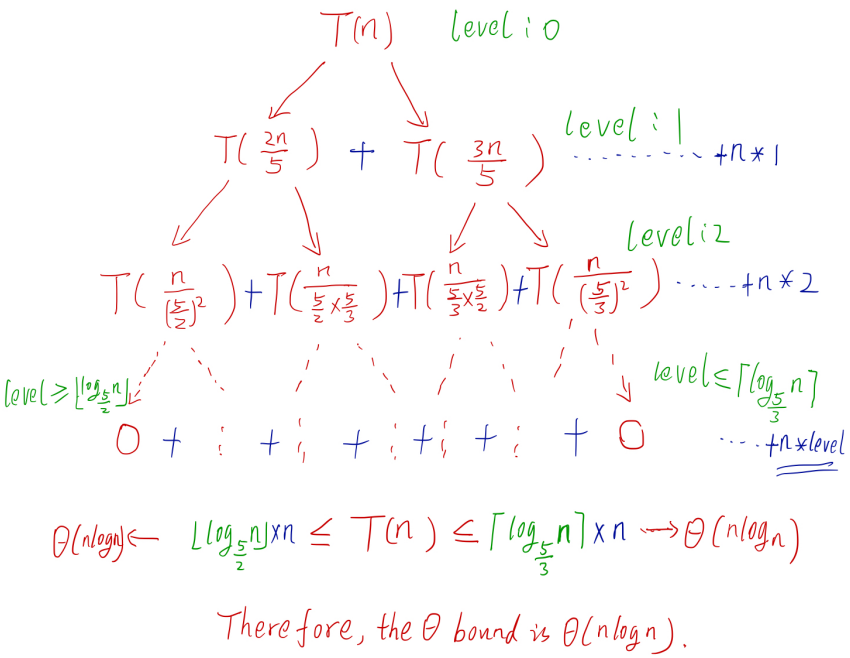


Figure 1: Recursion Tree

2. Use mathematical induction to show that $n! > 2^n$ for all $n \geq 4$, where n is a positive integer.

Solution:

Let $P(n)$ be the proposition that $n! > 2^n$ for all $n \geq 4$, where n is a positive integer.

Base case:
When $n = 4$, $P(4)$ is true as $4! = 24 > 2^4 = 16$.

Inductive step:
Assume that $P(k)$ is true, where k is some positive integers and $k \geq 4$, i.e. $k! > 2^k$

Now consider the case of $P(k+1)$,

$$\begin{aligned}
\text{L.H.S.} &= (k+1)! \\
&= k! \cdot (k+1) \\
&> 2^k \cdot (k+1) \\
&> 2^k \cdot 2 \\
&= 2^{k+1} \\
&= \text{R.H.S.}
\end{aligned}$$

Therefore, $P(k+1)$ is true.

According to M.I., $P(n)$ is true for all $n \geq 4$, where n is a positive integer.

3. In lecture, the sequence of Fibonacci numbers is discussed, i.e. $P(1) = P(2) = 1, P(n) = P(n-1)+P(n-2)$ for all $n \geq 3$. Use mathematical induction to prove that for all positive integers,

$$P(n) = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^n - (\frac{1-\sqrt{5}}{2})^n]$$

Solution:

Base case:
When $n = 1$, the statement is true since $1 = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^1 - (\frac{1-\sqrt{5}}{2}))^1]$
When $n = 2$, the statement is true since $1 = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^2 - (\frac{1-\sqrt{5}}{2}))^2]$

Inductive step:
Assume that the statement is true when $n = k$, where k is some positive integers and $k \geq 3$,
i.e. $P(k) = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^k - (\frac{1-\sqrt{5}}{2})^k]$

Now consider the case when $n = k + 1$,
L.H.S. $= P(k) + P(k-1)$

$$= \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^k - (\frac{1-\sqrt{5}}{2})^k] + \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^{k-1} - (\frac{1-\sqrt{5}}{2})^{k-1}]$$
$$= \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^{k-1}(\frac{3+\sqrt{5}}{2}) - (\frac{1-\sqrt{5}}{2})^{k-1}(\frac{3-\sqrt{5}}{2})]$$
$$= \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^{k-1}(\frac{1+2\sqrt{5}+5}{4}) - (\frac{1-\sqrt{5}}{2})^{k-1}(\frac{1-2\sqrt{5}+5}{4})]$$
$$= \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^{k-1}(\frac{1+\sqrt{5}}{2})^2 - (\frac{1-\sqrt{5}}{2})^{k-1}(\frac{1-\sqrt{5}}{2})^2)$$
$$= \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^{k+1} - (\frac{1-\sqrt{5}}{2})^{k+1}]$$
$$= \text{R.H.S.}$$

Therefore, the statement is true when $n = k + 1$.

According to M.I., the statement is true for all positive integers n.

4. The following pseudocode calculates the greatest common divisor of two positive integers `a` and `b`. Use M.I. to prove the correctness the gcd function.

```
gcd(a, b):
    if b == 0:
        return a
    else:
        return gcd(b, a mod b)
```

Solution:

If $b > a > 0$, we have $(a \texttt{ mod } b) = a$ and $\texttt{gcd}(a, b)$ returns $\texttt{gcd}(b, a)$. Hence, without loss of generality we can assume $a \geq b > 0$. Denote $p := \lfloor \frac{a}{b} \rfloor$ and $q := a - p \cdot b$, we have $q = (a \texttt{ mod } b)$ and $0 \leq q < b$.

Let $P(n)$ be the proposition that "when $b = n$, $\texttt{gcd}(a, b)$ returns the greatest common divisor of a and b".

(a) **Base Case.** Consider $n = 1$. When $b = 1$, $\texttt{gcd}(a, 1)$ returns $\texttt{gcd}(1, 0)$ which gives 1 and 1 is the greatest common divisor of a and 1.

(b) **Inductive step:** Assume that $P(k)$ is true for $k < n$, where n is some positive integer.
Now consider the case of $P(n)$,

- When $a = b$, we have $\texttt{gcd}(a, b)$ return $\texttt{gcd}(b, 0)$ which returns b and b is the greatest common divisor for a and b.
- When $a > b$, denote $r := \texttt{gcd}(a, b) = \texttt{gcd}(b, a \texttt{ mod } b)$. We show the followings:
  - r is a common divisor of a and b.
    As $(a \texttt{ mod } b) < b$, by induction hypothesis, r is the greatest common divisor of b and $(a \texttt{ mod } b)$, which implies that there exists some positive integers c and d such that $b = r \cdot c$ and $q = (a \texttt{ mod } b) = r \cdot d$. Hence $a = p \cdot b + q = p \cdot r \cdot c + r \cdot d = r \cdot (p \cdot c + d)$. Therefore r divides both a and b and the statement follows.
  - r is the greatest among all the common divisors of a and b.
    Let r' be the greatest common divisors of a and b, which implies that $r \leq r'$ and that there exists some positive integers m and n such that $a = r' \cdot m$ and $b = r' \cdot n$. Hence $(a \texttt{ mod } b) = q = a - p \cdot b = r' \cdot m - p \cdot r' \cdot n = r' \cdot (m - p \cdot n)$. Therefore r' divides both b and $(a \texttt{ mod } b)$ and we can conclude that r' is a common divisor of b and $(a \texttt{ mod } b)$ which implies that $r' \leq r$. Together with the fact that $r \leq r'$ we have $r = r'$, i.e., r is the greatest common divisor of a and b.

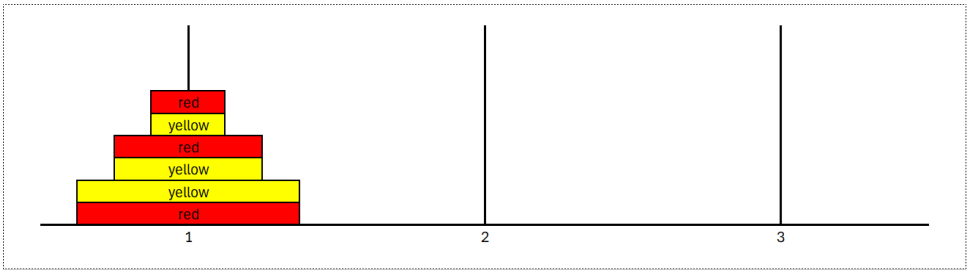
Therefore, $P(n)$ is true.

According to M.I., $P(n)$ is true for all positive integers n.

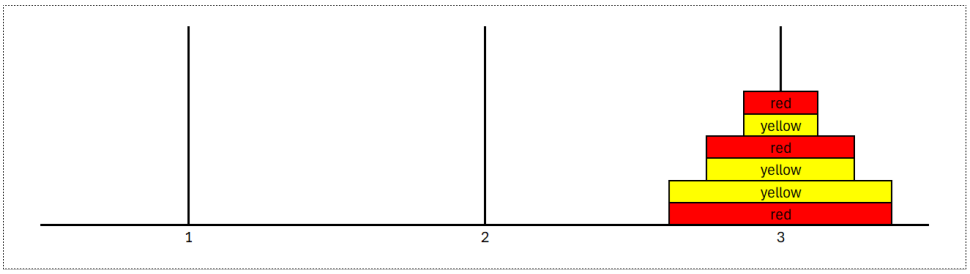
5. In lecture, the Towers of Hanoi problem is discussed. Consider the following modified version of the problem:

   There are n types of disk sizes and 3 pegs. For each of the disk size, there are one red disk and one yellow disk. All the disks are placed in order of size on the same peg. You are required to transfer the disks from the original peg to the destination peg in as few moves as possible, while keeping the original sequence of the disks. The constraints on moving the disks are the same as the original Towers of Hanoi problem mentioned in the lecture notes, i.e., no disk may be placed on top a disk that is smaller than it. For two disks of the same size, it is always safe to place one on top of the other regardless of their colours.

   Figure 2 shows the initial configuration and target configuration of an instance when $n = 3$.



(a) Initial Configuration



(b) Target Configuration

Figure 2: Initial Configuration and Target Configuration of an Instance when $n = 3$.

Let $P(n)$ be the number of moves required to move n types of disks for the problem above.

(a) Find $P(1)$, $P(2)$.

(b) Describe, in words, a recursive algorithm to solve the above-mentioned version of Towers of Hanoi.

(c) Hence, form a recurrence relation regarding to $P(n)$.

(d) Solve the recurrence relation obtained in part (c).

(e) Hence, give an upper bound for the time-complexity of the algorithm in part (b) in terms of big-O notation.

Solution:

(a) $P(1) = 3, P(2) = 11$

For $n = 1$:

1. Move the 1^{st} disk from the original peg to the buffer peg in 1 step.
2. Move the 2^{nd} disk from the original peg to the destination peg in 1 step.
3. Move the 1^{st} disk from the buffer peg to the destination peg in 1 step.

For $n = 2$:

1. Move the 1^{st} and 2^{nd} disk from the original peg to the destination peg in 2 steps (No need to consider their colour order).
2. Move the 3^{rd} and 4^{th} disk from the original peg to the buffer peg in 2 steps. Note that the colour order of these 2 disks are now reversed.
3. Move the 1^{st} and 2^{nd} disk from the destination peg to the original peg in 2 steps (No need to consider their colour order).
4. Move the 3^{rd} and 4^{th} disk from the buffer peg to the destination peg in 2 steps Their colour order are restored now.
5. Move the 1^{st} disk from the original peg to the buffer peg in 1 move.
6. Move the 2^{nd} disk from the original peg to the destination peg in 1 step.
7. Move the 1^{st} disk from the buffer peg to the destination peg in 1 step.

(b) Let $Q(n)$ be the number of moves involved in moving n types of disks (i.e. $2n$ disks) from the original peg to destination peg, without considering the colour order of the same-sized disks.

Consider moving the lowest 2 disks (With colour order preserved) from the source to the destination via the buffer peg. It can be done by:

1. Move the $2n-2$ disks from the original peg to the destination peg in $Q(n-1)$ steps.
2. Move the two largest disks from the original peg to the buffer peg in 2 steps. Note that the colour order of these 2 disks are now reversed.
3. Move the $2n-2$ disks from the destination peg to the original peg in $Q(n-1)$ steps.
4. Move the two largest disks from the buffer peg to the destination peg in 2 steps. The colour order of these 2 disks are correct now.

Repeat the steps above to move the remaining $2n-2$ disks in $P(n-1)$ steps. For the base case $n = 1$, it require 3 steps to move the disks.

(c) We first solve $Q(n)$. As it does not require preserving the original colour order of the disks, the number of steps required is equal to doubling the steps required by moving n disks in the original Tower of Hanoi problem, i.e. $Q(n) = 2(2^n - 1) = 2^{n+1} - 2$.

From the algorithm devised in part (b),

$$\begin{aligned} P(n) &= Q(n-1) + 2 + Q(n-1) + 2 + P(n-1) \\ &= (2^n - 2) + 2 + (2^n - 2) + 2 + P(n-1) \\ &= 2^{n+1} + P(n-1) \end{aligned}$$

(d) $P(n) = 2^{n+1} + P(n-1)$
$= 2^{n+1} + 2^n + ... + 2^3 + P(1)$
$= 2^3(2^{n-2} + 2^{n-3}... + 1) + P(1)$
$= 2^3(\frac{2^{n-1}-1}{2-1}) + 3$
$= 2^{n+2} - 5, \text{ for } n \geq 1$

(e) $O(2^n)$

Further Practice from the Textbook:
To gain more practice, here are a list of supplementary problems you may attempt from the course textbook **Introduction to Algorithms, 3rd edition**:

1. Solving Recurrence Relations:
   Exercise 2.3-3, 4.3-1, 4.3-2, 4.3-3, 4.3-6, 4-1, 4.3, 7.2-1, 7.4-1

2. Asymptotic Bounds:
   Exercise 3.1-1, 3.1-4, 3.2-5, 3.2-8, 3-2, 3-3, 3-4

3. Recursive Algorithm Design:
   Exercise 2.3-7, 9.3-8

However, due to copyright issues, we cannot directly upload the textbook or solution here. You may search the answers of these exercises online (E.g. By searching "Introduction to algorithms 3rd edition solution" on google).