School of Comp.
Sci. & Eng.

Homework No.1

April. 3, 2023

Notice

1. 문제는 특별한 이유가 없는 한, 손으로 풀어서 작성한다.
2. 작성된 리포트는 스캔 혹은 사진을 찍어서 하나의 압축된 파일로 묶은 후 PLATO 과제 제출란에 제출하거나 연구실 앞 과제 제출함에 제출한다.
연구실 : 자연대연구실험동 (건물번호 313동) 313호
3. Due Date : 4월 11일 24시

학과 : 정보컴퓨터공학부

학번 : 201924437

이름 : 김윤하

1. Given the recurrence relation
$T(n) = 7T(\frac{n}{5}) + 10n \qquad for \quad n > 1,$
$T(1) = 1$
find $T(625)$. **(Text Book Exercises 2.3 No: 14)**

$$T(625) = 7 \cdot T(\frac{625}{5}) + 10 \cdot 625$$
$$= 7 \cdot T(125) + 6250$$
$$T(125) = 7 \cdot T(\frac{125}{5}) + 10 \cdot 125$$
$$= 7\,T(25) + 1250$$
$$T(25) = 7 \cdot T(\frac{25}{5}) + 10 \cdot 25$$
$$= 7 \cdot T(5) + 250$$
$$T(5) = 7 \cdot T(\frac{5}{5}) + 10 \cdot 5$$
$$= 7\,T(1) + 50$$
$$= 57$$

$\Rightarrow T(25) = 7 \cdot 57 + 250 = 399 + 250 = 649.$

$\Rightarrow T(125) = 7 \cdot 649 + 1250 = 4543 + 1250 = 5793.$

$\Rightarrow T(625) = 7 \cdot 5793 + 6250 = 40551 + 6250 = 46801$

$\boxed{\therefore 46801}$

2. Write a divide-and-conquer algorithm for the Towers of Hanoi problem. The Towers of Hanoi problem consists of three pegs and n disks of different sizes. The object is to move the disks that are stacked, in decreasing order of their size, on one of the three pegs to a new peg using the third one as a temporary peg. The problem should be solved according to the following rules: (1) when a disk is moved, it must be placed on one of the three pegs; (2) only one disk may be moved at a time, and it must be the top disk on one of the pegs; and (3) a larger disk may never be placed on top of a smaller disk.

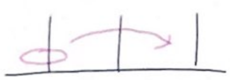
(a) Show for your algorithm that $\boxed{S(n) = 2^n - 1}$. (Here $S(n)$ denotes the number of steps (moves), given an input of n disks.)

(b) Prove that any other algorithm takes at least as many moves as given in part (a).
→ 하노이 탑 문제에서 최적의 해결책이 유일함

(Text Book Exercises 2.3 No: 17)

(a) 원반 중 가장 큰 것을 제외한 $n-1$ 개의 원반을 중간 막대로 이동시키고,
마지막 가장 큰 원반을 마지막 막대로 이동시킨 후, 중간 $n-1$ 개를 마지막 막대로 옮기는
재귀적 알고리즘으로 구현이 가능하다.

i) $n=1$ 인 경우,

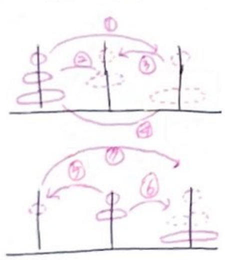


1 번으로 가능하다. $\Rightarrow n=1 ; S(n) = 1 = 2^1 - 1$

ii) $n=2$ 인 경우,

3 번으로 가능하다. $\Rightarrow n=2 ; S(n) = 3 = 2^2 - 1$

iii) $n=3$인 경우



7 번으로 가능하다 $\Rightarrow n=3 ; S(n) = 7 = 2^3 - 1$.

iv) $n=4$ 인 경우 : 15 번으로 가능하다. $\Rightarrow n=4 ; S(n) = 15 = 2^4 - 1$

v) $n=5$ 인 경우, 31번으로 가능하다. $S(5) = 31$
$= 2^5 - 1$

$$\boxed{\therefore\ S(n) = 2^n - 1}.$$

6) 재귀적으로 문제를 해결하는 상황에서,
n 개의 원판 중, $n-1$ 개를 2번째 막대로 옮겨야 하고,
나머지 가장 큰 원판을 3번째 막대로 옮긴다.
이 때 $n-1$ 개를 정리할 때도 마찬가지로,
$n-2$ 개는 다른 막대로 옮기고, (1st 막대)
$n-1$ 번째를 3 번째 막대로 옮기는 것을 알 수 있다.
따라서, $(n-1)$ 개 정리를 2 번 하고, 마지막 막대 옮기는 것을 1 번 한다고 볼 수 있다.
여기서 $S(n) = 2 * S(n-1) + 1$ 임을 알 수 있다. 이를 정리하면,

$$\begin{aligned}
S(n) &= 2 \cdot S(n-1) + 1 \\
&= 2 \cdot (2S(n-2)+1) + 1 \\
&= 4S(n-2) + 2 + 1 \\
&= 2 \cdot (4S(n-3)+2+1) + 1 \\
&= 8S(n-3) + 4 + 2 + 1 \\
&\ \ \vdots \\
&= 2^{n-1}\overset{=1}{S(1)} + 2^{n-2} + 2^{n-3} + \cdots + 2 + 1 \\
&= 2^{n-1} + 2^{n-2} + 2^{n-3} + \cdots + 2 + 1 \\
&= 2^n - 1 .
\end{aligned}$$

$(a=1,\ r=2,\ n$ 인 수열$) \longrightarrow \dfrac{2^n - 1}{2-1} = 2^n - 1$

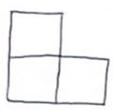
$$\boxed{\therefore\ S(n) = 2^n - 1.}$$

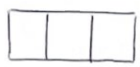
3. A tromino is a group of three unit squares arranged in an L-shape Consider the following tiling problem: The input is an $m \times m$ array of unit squares where m is a positive power of 2, with one forbidden square on the array. The output is a tiling of the array that satisfies the following conditions:

- Every unit square other than the input square is covered by a tromino.
- No tromino covers the input square.
- No two trominos overlap.
- No tromino extends beyond the board.

Write a divide-and-conquer algorithm that solves this problem. **(Text Book Chap. 2 Additional Exercises No: 42)**

i) 트로미노의 두 가지 경우 ;

①


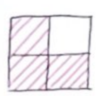
②


에서 ① 이 L-shape tromino 이다.

ii) 접근 방법.

function Tile (n , m) 에서 ,

① tile 이 1개인 경우 , 규칙을 만족시키는 자명한 해가 존재한다 .



와 같이 만족시키므로 , terminate 한다.

② 1이 아닌 경우 , 보드를 4 개의 $\frac{n}{2} \times \frac{n}{2}$ 의 하위 타일보드로 나누고 ,
그 중 세 개의 사각형이 m 을 포함하지 않는 3개의 하위 보드에 각각 하나의 tromino 놓는다.

③ 각각 하위 보드에 놓인 tromino 가 되는 사각형의 위치를 m1, m2, m3, m4로 하고,
각각의 4가지 경우에 대해, 다시 Tile 함수를 호출한다.

iii) 재귀적 Algorithm 으로 구현.

1) $n=2$일 경우, 2x2 로 자명함 (한 가지 경우)

2) 4가지 경우에 각각의 사분면에 대해

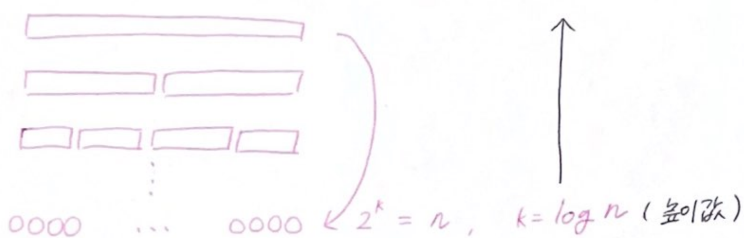
Tile ($\frac{n}{2}$, m1) ; Tile ($\frac{n}{2}$, m3) ;
Tile ($\frac{n}{2}$, m2) ; Tile ($\frac{n}{2}$, m4) ; 호출해준다.

4. Use the divide-and-conquer approach to write a recursive algorithm that finds the maximum sum in any contiguous sublist of a given list of n real values. Analyze your algorithm, and show the results in order notation. **(Text Book Chap 2. Additional Exercises No: 45)**

→ 최대 연속 부분합 구하기.

1. input 으로 List A, 크기 n 을 받는다.
2. if $n == 1$ 이면 하나의 원소 값을 return 해준다.
3. if $n > 1$ 이면, List 를 두 개로 분할한다.
4. left subList 의 최대 연속 부분합을 찾고, right subList 도 마찬가지로 찾는다.
5. 중앙 부분을 고려한 최대 연속 부분합을 계산한다. (왼쪽, 오른쪽)
6. 위의 4, 5 과정에서 구한 세가지 중 최대값을 return 해준다.

i) → List 를 반으로 나누는 데 걸리는 시간 $O(\log n)$



ii) → 각 리스트에서 최대 연속 부분합을 구하는 데 걸리는 시간 $O(n)$

iii) 합산한 최대 연속 부분합의 시간 복잡도 = $O(n \cdot \log n)$ 이다.

$\boxed{\therefore O(n \log n)}$

5. Use induction on n to show that the divide-and-conquer algorithm for the Binomial Coefficient problem (Algorithm 3.1), based on Equality 3.1, computes $2\binom{n}{k}-1$ terms to determine $\binom{n}{k}$. **(Text Book Exercises 3.1 No: 2)**

$(x+y)^n = \sum_{k=0}^{n} \frac{n!}{k!(n-k)!} x^k y^{n-k}$ 이다.

↳ i) $n=1$ 인경우,

if) $k==0$ 이면

$$2\binom{n}{k}-1 = 2\cdot 1 - 1 = 1.$$

ii) $k=1$, $k=n$ 인 경우,

$$2\binom{n}{k}-1 = 2n-1 = 2-1 = 1.$$

⇒ base case 는 "true"

↳ $k=0$, $k=n$ 인 경우는 쉽다.

일반적인 k 에 대해 고려해보면, return $C(n-1, k-1) + C(n-1, k)$ 해주면 된다.

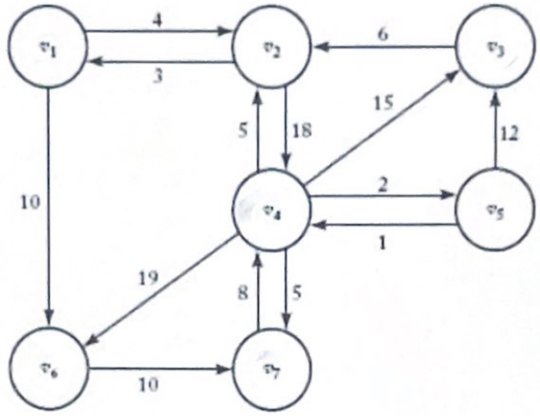
$$2\binom{n}{k}-1 \Rightarrow 2\binom{n}{k-1}-1 + 2\binom{n}{k}-1+1$$

$$= 2\left(\binom{n}{k-1}+\binom{n}{k}\right)-1$$

↳ $n+1$ 인 경우를 생각해보면,

$\binom{n+1}{k}$ 는 $2\binom{n+1}{k}-1$ 의 항을 갖는다.

6. Use the Print Shortest Path algorithm (Algorithm 3.5) to find the shortest path from vertex v_7 to vertex v_3, in the graph of Exercise 5, using the matrix P found in that exercise. Show the actions step by step.



(Text Book Exercises 3.2 No: 6)

matrix P (바로 연결된 것들만 고려)

=

	V_1	V_2	$\boxed{V_3}$	V_4	V_5	V_6	V_7
V_1	0	4				10	
V_2	3	0		18			
V_3		6	0				
V_4		5	15 ✓	0	2 ✓	19	5
V_5			12 ✓	1	0		
V_6						0	10
$\boxed{V_7}$				8 ✓			0

위에서, $V_7 \to V_3$ 의 방법은

$V_7 \to V_4 \to V_3$, $V_7 \to V_4 \to V_5 \to V_3$ 의 두 가지이다.

① $V_7 \to V_4 \to V_3$ 의 경우, $8 + 15 = 23$.

② $V_7 \to V_4 \to V_5 \to V_3$ 의 경우, $8 + 2 + 12 = 22$.

① > ② 이므로, ② $V_7 \to V_4 \to V_5 \to V_3$ 을 택한다.

$\boxed{\therefore V_7 \to V_4 \to V_5 \to V_3 \ (22)}$

7. List all of the different orders in which we can multiply five matrices A, B, C, D, and E. **(Text Book Exercises 3.4 No: 12)**

m×n matrix 를 곱할 때,

$M_1 = a \times b$, $M_2 = c \times d$ 이면

$b = c$ 이어야 한다. ($M_1 \times M_2$ 가능)

따라서 A와 B, B와 C, C와 D, D와 E 에서의 $b=c$ 라 하면,

(그래야 곱할 수 있음) 가능한 곱의 경우는 다음과 같다.

① ((((A B) C) D) E)

② (((A B) C) (D E))

③ ((A B)(C (D E)))

④ ((A B) C (D E))

⑤ ((A B)(C D) E)

⑥ (A (B C)(D E))

⑦ (A (B C) D E)

⑧ (A (B (CD)) E)

⑨ (A (B (C (D E))))

⑩ (A (B (C D) E))