CS 2500 - ALGORITHMS

#1

SOLUTION TO HW #1

Problem 1.1

(i) Consider the Taylor Series expansion of e^x.

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = \sum_{i=0}^{m-1} \frac{x^i}{i!} + \sum_{i=m}^{\infty} \frac{x^i}{i!}$$

Note: If $\sum_{i=m}^{\infty} \frac{x^i}{i!} = \Theta(x^m)$, the proof is complete.

Now: $$\sum_{i=m}^{\infty} \frac{x^i}{i!} = x^m \cdot \sum_{i=m}^{\infty} \frac{x^{i-m}}{i!}$$

$$= \frac{x^m}{m!} \underbrace{\left[1 + \frac{x}{m+1} + \frac{x^2}{(m+1)(m+2)} + \cdots \right]}_{\text{let this be } g(x).} \quad \text{— (1)}$$

Note that since $x \in [0, 1]$,

$$g(x) \geq g(0) = 1. \quad \left(\text{since every term } \frac{x^i}{(m+1)\cdots(m+i)} \geq 0. \right) \quad \text{— (2)}$$

$|||^{ly}$, $$g(x) \leq 1 + \frac{x}{m+1} + \frac{x^2}{(m+1)^2} + \frac{x^3}{(m+1)^3} + \cdots$$

since $\frac{1}{m+1} > \frac{1}{m+i} \quad \forall\, i > 1.$

#2

In other words, $g(x) \le 1 + \dfrac{x}{m+1} + \dfrac{x^2}{(m+1)^2} + \cdots$

$$\le 1 + \frac{1}{m+1} + \frac{1}{(m+1)^2} + \cdots$$

Since $x \in [0,1]$

$$= \frac{1}{1 - 1/_{m+1}} = \frac{m+1}{m}. \quad \text{③}$$

$\therefore$ Combining equations ② and ③,

$$1 \le g(x) \le \frac{m+1}{m}.$$

Substituting this in eqn. ①, we have

$$\frac{x^m}{m!}(1) \le \sum_{i=m}^{\infty} \frac{x^i}{i!} \le \frac{x^m}{m!}\left(\frac{m+1}{m}\right)$$

$$\Rightarrow \sum_{i=m}^{\infty} \frac{x^i}{i!} = \Theta(x^m) \quad \square.$$

(#3)

(ii)

$$\log(n!) = \log 1 + \log 2 + \cdots + \log n$$

Since log is a monotonically increasing fn.

$$\leq \log n + \cdots + \log n$$

$$= n \cdot \log n.$$

$$\Rightarrow \log n! = O(n \log n). \quad \text{— (1)}$$

$$\log n! = \log 1 + \log 2 + \cdots + \log n$$

Considering the last half of the sum.

$$\geq \log\left(\frac{n}{2}\right) + \log\left(\frac{n}{2}+1\right) + \cdots + \log n$$

monotonicity of log.

$$\geq \log\left(\frac{n}{2}\right) + \log\left(\frac{n}{2}\right) + \cdots + \log\left(\frac{n}{2}\right)$$

$$= \left(\frac{n}{2}\right) \cdot \log\left(\frac{n}{2}\right)$$

$$= \frac{n}{2} \log n - \frac{n}{2} \log 2$$

Since $n \geq 4$

$\log n \geq 2 \log 2$

$\Rightarrow \frac{n}{4} \log n \geq \frac{n}{4}(2 \log 2) = \frac{n}{2} \log 2$

$$\geq \frac{1}{4} \cdot n \log n \quad \forall\, n \geq 4.$$

$$\Rightarrow \log n! = \Omega(n \log n) \quad \text{— (2)}$$

Combine (1) & (2) $\Rightarrow \log(n!) = \Theta(n \log n)$

Solutions to HW1

Problem 1.2

3.1

Let $p(n) = \sum_{i=0}^{d} a_i n^i$

(a) For any $k \geq d$, we have

$$\frac{p(n)}{n^k} = \sum_{i=0}^{d} a_i n^{i-k}$$

Note that all these terms have negative powers.

Choose any $n_0 \in \mathbb{N}$.

Then, for any $n \geq n_0$,

$$\frac{p(n)}{n^k} \leq \sum_{i=0}^{d} a_i n_0^{i-k}$$

Therefore $p(n) \leq c \cdot n^k \quad \forall \quad n \geq n_0$

where $c = \sum_{i=0}^{d} a_i n_0^{i-k}$.

i.e. $p(n) = O(n^k)$.

(b) If $k \leq d$, then

$$\frac{p(n)}{n^k} = \sum_{i=0}^{d} a_i n^{i-k}$$

$$= \sum_{i=0}^{k-1} a_i n^{i-k} + a_k + \sum_{i=k+1}^{d} a_i n^{i-k}$$

$\therefore \quad \dfrac{p(n)}{n^k} \geq a_k$ for all $n \in \mathbb{N}$.

$\therefore \quad p(n) \geq a_k \cdot n^k \quad \forall \; n \geq 1$

$\Rightarrow \quad p(n) = \Omega(n^k)$.

(c) If $k = d$, then

$$\frac{p(n)}{n^k} = \sum_{i=0}^{k} a_i n^{i-k}$$

$$= \sum_{i=0}^{k-1} a_i n^{i-k} + a_k .$$

Our goal is to prove that

$\exists\, n_0 \in \mathbb{N}$ s.t. $c_1 n^k \le p(n) \le c_2 n^k$ for all $n \geq n_0$

i.e. $$c_1 \le \frac{p(n)}{n^k} \le c_2 \quad \text{for all } n \geq n_0 .$$

i.e. $$c_1 \le \sum_{i=0}^{k-1} a_i n^{i-k} + a_k \le c_2 \quad \forall\, n \geq n_0 .$$

Choose $c_1 = a_k$, $c_2 = \sum_{i=0}^{k-1} a_i n_0^{i-k} + a_k$.

This choice holds true for any $n_0 \in \mathbb{N}$.

$\therefore\ p(n) = \Theta(n^k)$.

(d) If $k > d$,

$$\frac{p(n)}{n^k} = \sum_{i=0}^{d} a_i n^{i-k}$$

all the terms have strictly negative exponents.

Therefore, for any $n_0 \in \mathbb{N}$,

we have

$$\sum_{i=0}^{d} a_i n^{i-k} < \sum_{i=0}^{d} a_i n_0^{i-k}$$

for all $n \geq n_0$.

Therefore, if $c = \sum_{i=0}^{d} a_i n_0^{i-k}$, we have

$$p(n) < c \cdot n^k \quad \text{for all } n \geq n_0.$$

$$\Rightarrow \quad p(n) = o\left(n^k\right).$$

e) If $k < d$,

$$\frac{p(n)}{n^k} = \sum_{i=0}^{k-1} a_i n^{i-k} + a_k + \sum_{i=k+1}^{d} a_i n^{i-k}$$

this term now is strictly positive.

Since $k \neq d$ and $k < d$,

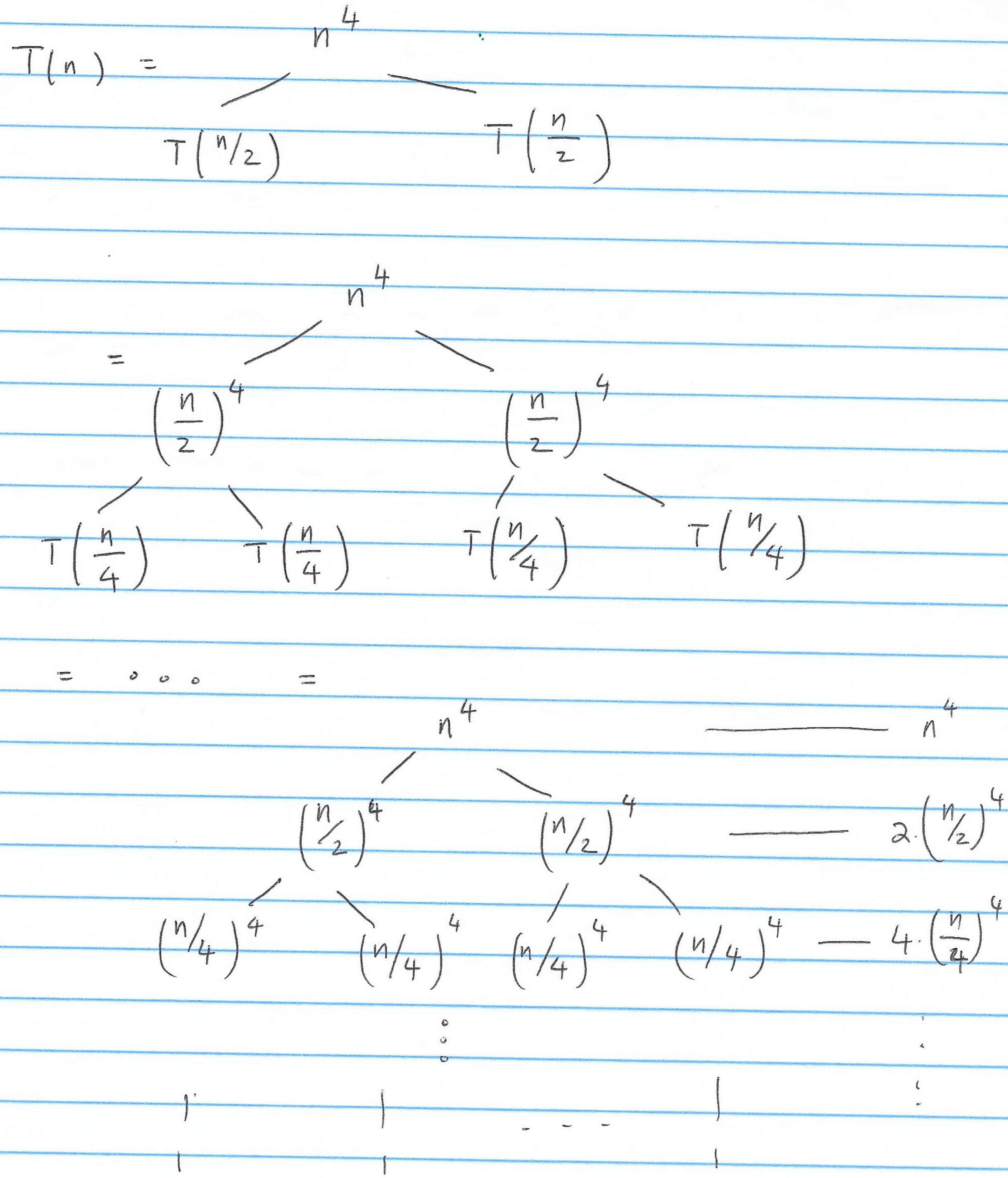
$$\frac{p(n)}{n^k} > a_k \text{ for all } n \in \mathbb{N}.$$

$$\Rightarrow \quad p(n) > a_k n^k \quad \forall \; n \geq 1.$$

i.e. $p(n) = \omega(n^k)$.

Problem 4.1

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



$\therefore$ The sum of all the levels in the recursion tree are given by

$$T(n) = \sum_{k=0}^{L-1} 2^k \cdot \left(\frac{n}{2^k}\right)^4 = n^4 \cdot \sum_{k=0}^{L-1} \frac{1}{2^{3k}}$$

$$= n^4 \sum_{k=0}^{L-1} \left(\frac{1}{8}\right)^k = n^4 \cdot \left(\frac{1-\left(1/8\right)^L}{1-1/8}\right)$$

$$= \frac{8n^4}{7}\left(1-\left(\frac{1}{8}\right)^L\right).$$

Here, L is the # levels in the recursion tree.

$\therefore$ At the boundary condition, we have

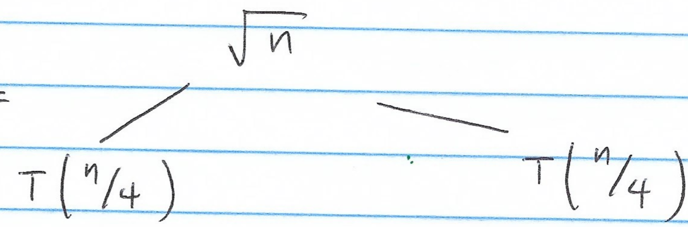
$$n/2^L = 1 \quad \text{i.e.} \quad L = \log_2 n.$$

$$\therefore \quad T(n) = \frac{8n^4}{7}\left[1-\left(\frac{1}{8}\right)^{\log_2 n}\right]$$

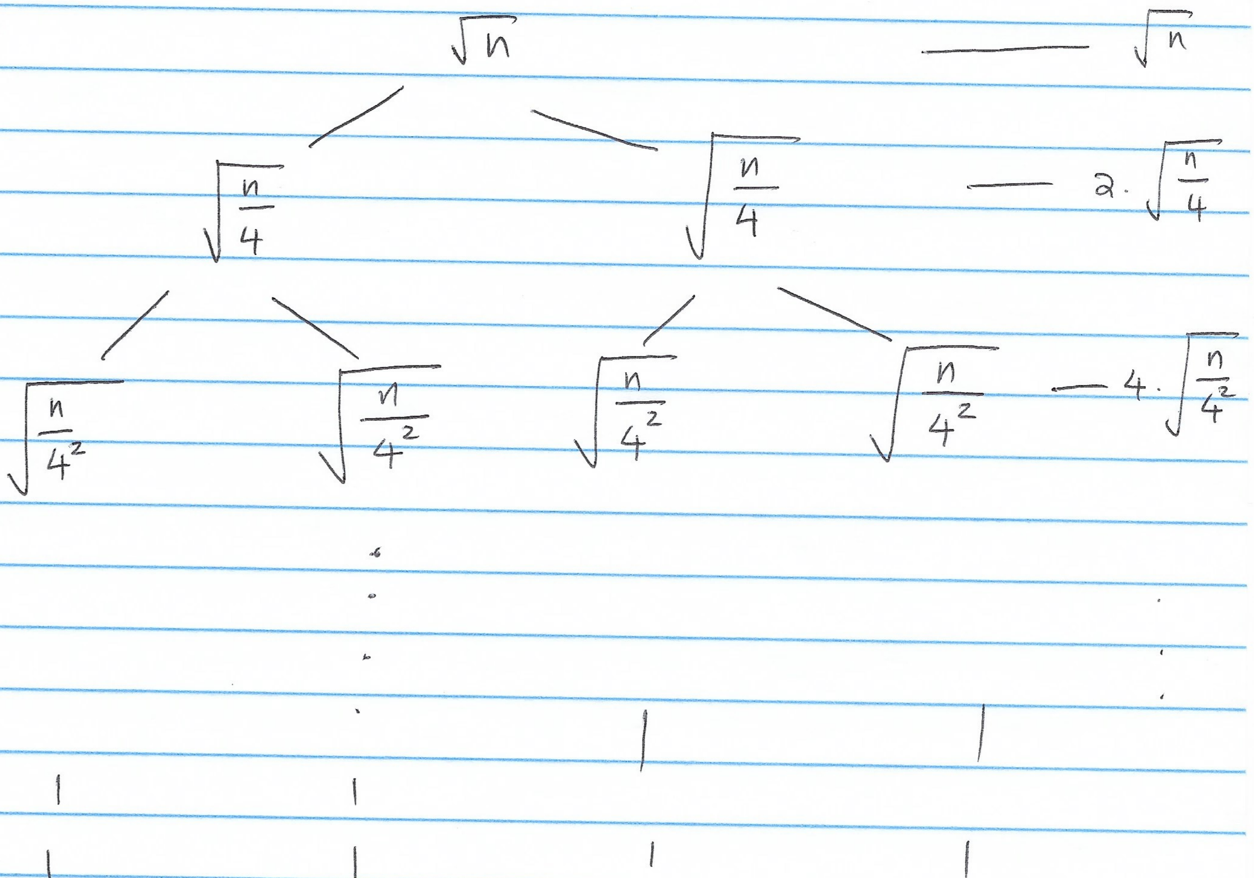
$$= \frac{8n^4}{7}\left[1-\frac{1}{n^3}\right] = \frac{8n(n^3-1)}{7}$$

$$= \Theta(n^4)$$

(f) $T(n) = 2T(n/4) + \sqrt{n}$

$T(n) =$ 

$= \ \circ\circ\circ \ =$



$$\therefore \quad T(n) = \sum_{i=0}^{L-1} 2^i \sqrt{\frac{n}{4^i}} = \sum_{i=0}^{L-1} 2^i \cdot \frac{\sqrt{n}}{2^i} = L \cdot \sqrt{n}$$

P.T.O.

The # levels in the recursion tree are

$$L = \log_4 n \quad \left(\text{since } \frac{n}{4^L} = 1 \text{ at termination} \right).$$

$$\Rightarrow \quad T(n) = \Theta(n \log n).$$

(g) $$T(n) = T(n-2) + n^2$$

$$= \left[T(n-4) + (n-2)^2 \right] + n^2$$

$$= \left[\left\{ T(n-6)^2 + (n-4)^2 \right\} + (n-2)^2 \right] + n^2$$

$$= \begin{cases} 1^2 + 3^2 + \cdots + n^2 & \text{if } n \text{ is odd} \\ 2^2 + 4^2 + \cdots + n^2 & \text{if } n \text{ is even.} \end{cases}$$

$$\cancel{= \frac{n(4n^2-1)}{3} \text{ if } n \text{ is odd}}$$

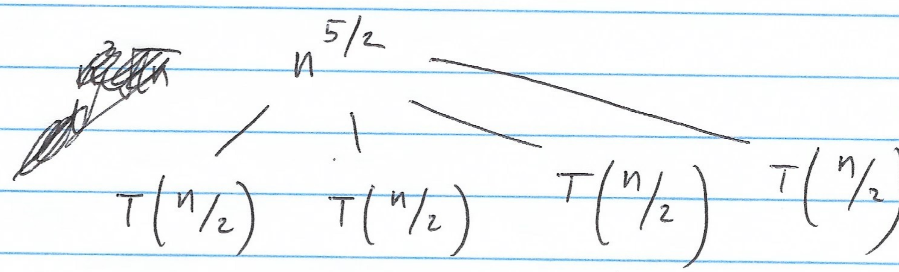
$$= \begin{cases} \dfrac{n(n-1)(n-2)}{6} & \text{if } n \text{ is odd.} \\ \dfrac{n(n+1)(n+2)}{6} & \text{if } n \text{ is odd.} \end{cases}$$

Either way, $T(n) = \Theta(n^3)$.

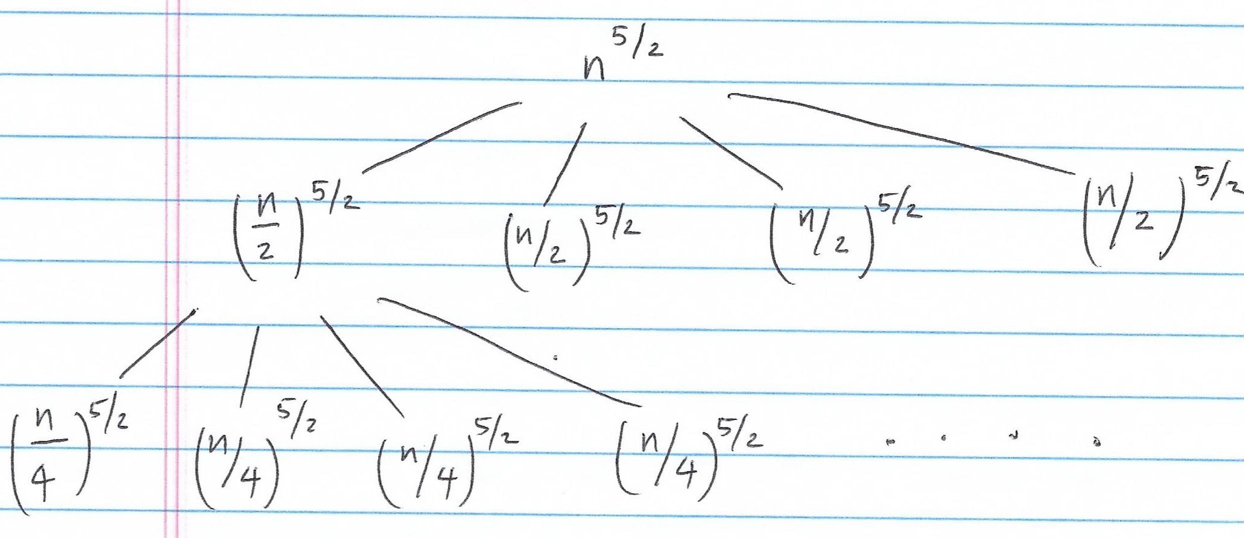
Problem 4.3

(c) $T(n) = 4T(n/2) + n^2\sqrt{n}$

$T(n) =$



$= \quad \ldots \quad =$



$$\therefore\ T(n) = \sum_{i=0}^{L-1} 4^i \cdot \left(\frac{n}{2^i}\right)^{5/2} = n^{5/2} \cdot \sum_{i=0}^{L-1} \left(\frac{1}{\sqrt{2}}\right)^i$$

P.T.O.

#14

$$\therefore \quad T(n) = n^{5/2}\left(\frac{1-\left(1/\sqrt{2}\right)^{L}}{1-1/\sqrt{2}}\right)$$

where $L = \log_2 n$.

$$\therefore \quad T(n) = n^{5/2} \cdot \frac{\left(1 - 1/n^2\right)}{1-1/\sqrt{2}}$$

$$= \frac{\sqrt{n}\left(n^2-1\right)}{\left(\sqrt{2}-1\right)} \cdot \sqrt{2} = \Theta\left(n^{5/2}\right)$$

(h) $T(n) = T(n-1) + \log n$

$$= \left[T(n-2) + \log(n-1)\right] + \log n$$

$$= \left[\left\{T(n-3) + \log(n-2)\right\} + \log(n-1)\right] + \log n$$

$$= \cdots = \log n + \log(n-1) + \cdots + \log 1$$

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

#15

(j) $T(n) = \sqrt{n}\, T(\sqrt{n}) + n$

The recursion tree is given by

$$T(n) = \begin{array}{c} n \\ | \\ \sqrt{n}\, T(\sqrt{n}) \end{array} = \begin{array}{c} n \\ | \\ \sqrt{n} \left(\begin{array}{c} \sqrt{n} \\ | \\ T(n^{1/4}) \cdot n^{1/4} \end{array} \right) \end{array}$$

$$= \begin{array}{c} n \\ | \\ n \\ | \\ n^{3/4} \cdot T(n^{1/4}) \end{array} = \cdots = \begin{array}{c} n \\ | \\ n \\ | \\ n \\ | \\ \vdots \end{array}$$

$\therefore \quad T(n) = L \cdot n$

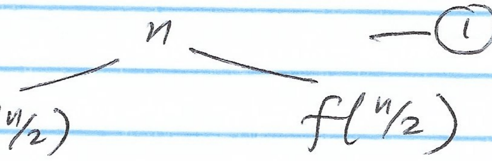
where L = # of levels in the recursion tree.

If the recursion terminates at $n^{1/2^L} = 1$, and if $n \in \mathbb{R}$, then $L = \infty$. $\Rightarrow T(n) = \infty$.

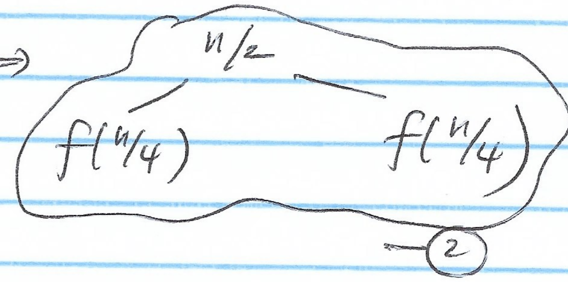
Prob. 1.3

Recursion tree for $f(n) = 2f\left(\frac{n}{2}\right) + n$

$f(n) = 2 \cdot f(n/2) + n \Rightarrow$

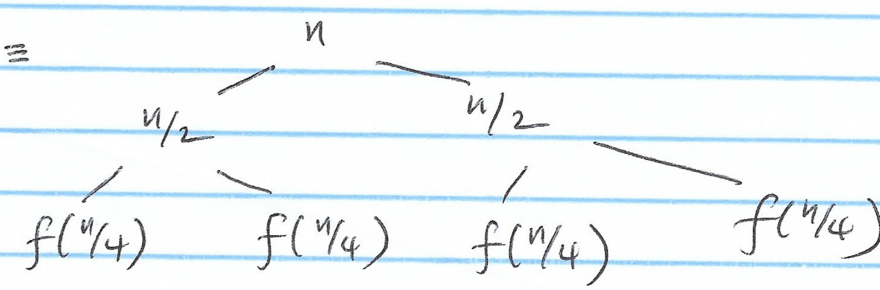


$f(n/2) = 2 \cdot f(n/4) + n/2 \Rightarrow$



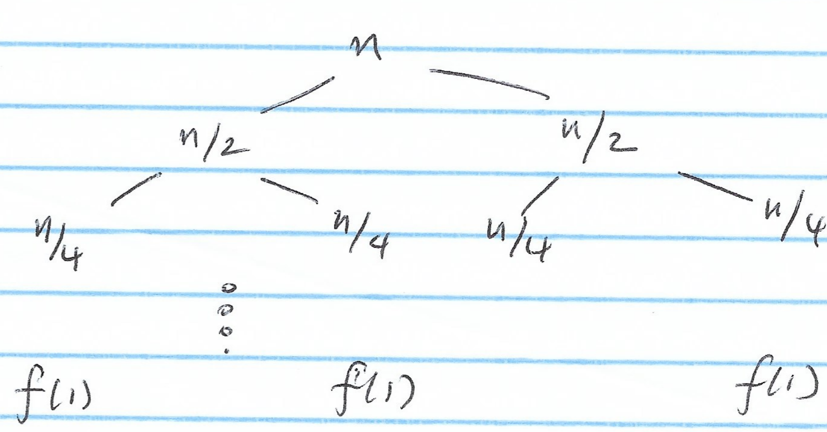
Replacing the leaves in 1 with 2, we get

$f(n) \equiv$



Repeating the same for $f(n/4)$, $f(n/8)$, ..., $f(n/2^i)$,

we obtain



#17

Note that the sum of all terms in level-$i = n$. $\forall\ i = 1, 2, \ldots, k-1$

where k is the final level.

$$\therefore\ f(n) = (k-1).\,n + 2^k\left[f(1)\right]$$

At the final level, note that $\frac{n}{2^k} = 1$.

$$\Rightarrow 2^k = n. \Rightarrow k = \log_2 n$$

$$\therefore\ f(n) = n(\log n - 1) + n\,(f(1))$$

$$= \Theta(n \log n)$$

Problem 3.1

Proof of correctness for Algorithm A

Let us choose the following loop-invariant.

At the $(i^*, j^*)^{th}$ iteration,

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \quad \text{for all } i = 1, \ldots, i^* \text{ and } j = 1, \ldots, j^*.$$

Base case: After $(1,1)^{th}$ iteration,

$$c_{11} = \sum_{k=1}^{n} a_{1k} b_{k1}$$ holds true since

before the for loop starts ticking the index k, we initiate $c_{11} = 0$ and increment c_{11} every time with $a_{1k} \cdot b_{k1}$ for different values of k, for all $k = 1, \ldots, n$.

Intermediate case: Say, the loop invariant holds true for all (i, j) such that $i = 1, \ldots, i^*$ and $j = 1, \ldots, j^*$.

The next iteration has two possibilities:

P.T.O.

<u>Case-1</u>: The next iteration is (i^*, j^*+1) if $j^* \neq n$.

<u>Case-2</u>: The next iteration is $(i^*+1, 1)$ if $j^* = n$.

In both the cases, after the iteration,

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \quad \text{for both } (i,j) = \begin{cases} (i^*, j^*+1) \\ (i^*+1, 1) \end{cases}$$

due to the same reason as presented in base case.

<u>Terminal case</u>: Therefore, for $(i,j) = (n,n)$

Since the base case and intermediate cases hold true,

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \quad \forall (i,j) = (1,1), \cdots, (n,n).$$

Proof of correctness of Algorithm B

$$C = AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$\therefore \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$$

But, to compute the ~~prog~~ product $A_{ij}B_{kl}$, we perform another recursion call.

Let us prove this formally by induction.

BASE CASE: If A, B are 1×1 matrices, we have $C = AB$.

If A, B are 2×2 matrices,

$$C = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} \\ \\ \end{bmatrix}$$

#21

Intermediate case:

Say, the algorithm already completed the ~~4~~ matrices

$$A_{11},\ A_{12},\ A_{21},\ A_{22}$$

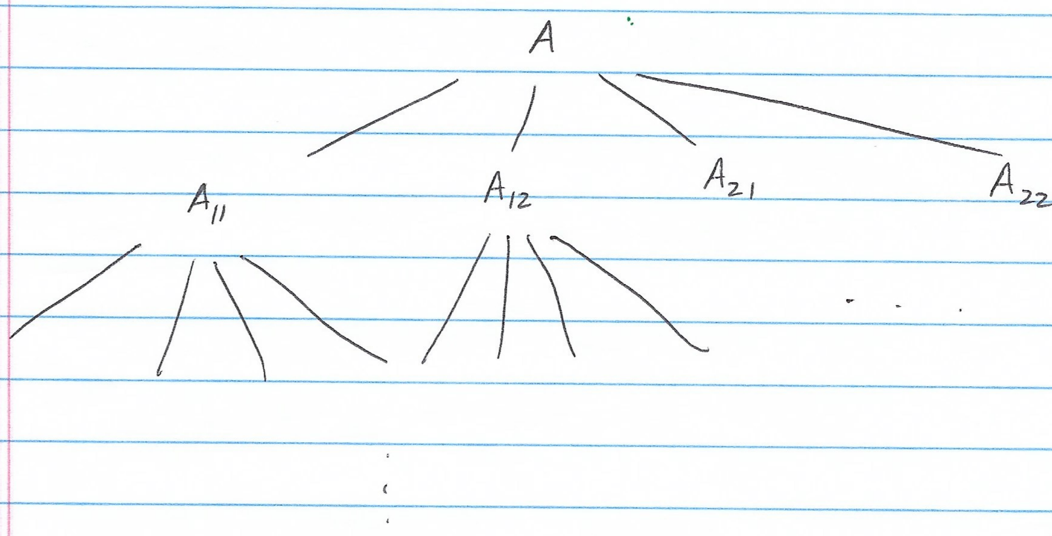
$$B_{11},\ B_{12},\ B_{21},\ B_{22}$$

using the recursion method, after k iterations

where k

Proof of correctness of Algorithm B

Consider a matrix A. Algorithm B splits A as shown below:



This is recursively partitioned until we end up with scalar terms.

Therefore, we consider the scalar terms as our base case in our induction proof.

Base case: Given two matrices A, B both of size 1×1, we have

$$C = [a][b] = ab.$$

P.T.O.

Proof of correctness of Algorithm B

Note that

$$P = AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$= \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$$

has 8 products.

$$P_1 = A_{11}B_{11}, \quad P_2 = A_{12}B_{21}$$

$$P_3 = A_{11}B_{12}, \quad P_4 = A_{12}B_{22}$$

$$P_5 = A_{21}B_{11}, \quad P_6 = A_{22}B_{21}$$

$$P_7 = A_{21}B_{12}, \quad P_8 = A_{22}B_{22}$$

Since the algorithm

Proof of correctness of Algorithm B

Let $P_0 = AB$

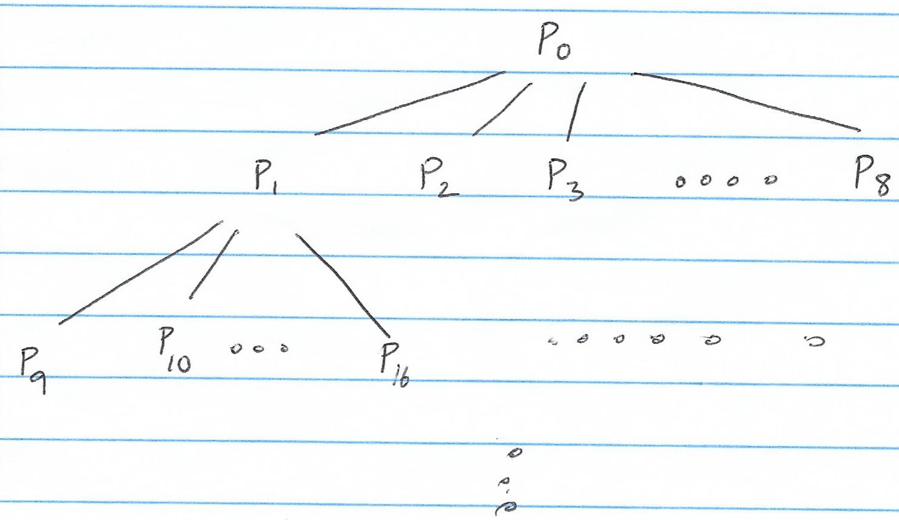
$$= \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}.$$

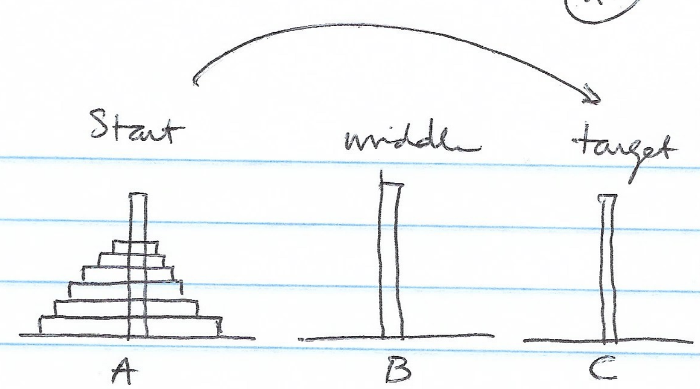
Note that it has 8 products. If $P_1 = A_{11}B_{11}$,

$P_2 = A_{12}B_{21}$, $P_3 = A_{11}B_{12}$, $P_4 = A_{12}B_{22}$, $P_5 = A_{21}B_{11}$,

$P_6 = A_{22}B_{21}$, $P_7 = A_{21}B_{12}$ and $P_8 = A_{22}B_{22}$,

we have





Problem 4

4.1 Tower (n, A, C, B)

if $n \neq 0$

Tower $(n-1, A, B, C)$ ← moves $(n-1)$ top disks from A to B.

move n^{th} disk from A to C.

Tower $(n-1, B, C, A)$ ← moves $(n-1)$ disks from B to C.

Claim: Tower( ) minimizes the sequence of moves needed to transfer n pegs from A to C.

Proof: Let $M_{AC}(D_1, \ldots, D_n)$ denote the min. sequence for moving disks $D_1, \ldots, D_n$.

We prove this claim using principles of induction.

Base case: When $n = 1$, $M_{AC}(D_1) = \{$ move D_1 from A to C $\}$.
← cannot do any faster.

~~Intermediate case:~~
~~Assume $M(D_1, \ldots, D_k)$ is the min. sequence of moving k~~

But, $n=1$ is not our base case since $n-1=0$.

Base case $\Rightarrow$ $n = 2$.

$$\mathcal{M}_{AC}(D_1, D_2) = \{ \mathcal{M}_{AB}(D_1), \mathcal{M}_{AC}(D_2), \mathcal{M}_{BC}(D_1) \}$$

min. # of moves with two disks.
(cannot do any faster)

Intermediate case: Assume $\mathcal{M}_{AC}(D_1, \ldots, D_k)$ is the min. sequence.

Say, our goal is to move $(k+1)$ disks.

$\Rightarrow$ Partition $(D_1, \ldots, D_{k+1})$ into two super disks:

$$S_1 = (D_1, \ldots, D_k)$$

$$S_2 = D_{k+1}.$$

$$\Rightarrow \mathcal{M}_{AC}(S_1, S_2) = \{ \mathcal{M}_{AB}(S_1), \mathcal{M}_{AC}(S_2), \mathcal{M}_{BC}(S_1) \}$$

(from base case)

min. sequence to move (S_1, S_2) from A to C.
(two superdisks.)

Termination: At the terminal stage, all n disks would be moved and algorithm terminates. $\square$.

4.2 Run-time

$$T(n) = \underbrace{2 \cdot T(n-1)}_{\text{time to move } (n-1) \text{ pegs}} + \underbrace{c}_{\text{time to move one peg}}$$

Claim: $T(n) = \Theta(2^n)$

Proof: $T(n) = 2 \cdot T(n-1) + c$

$$= 2 \cdot \left[2 \cdot T(n-2) + c \right] + c$$

$$= 4 \cdot T(n-2) + 2 \cdot c + c$$

$$= 4 \cdot \left[2T(n-3) + c \right] + 2c + c$$

$$= 8 \cdot T(n-3) + 4 \cdot c + 2 \cdot c + c$$

$$= 8 \left[2 \cdot T(n-4) + c \right] + 4c + 2c + c$$

$$= 16 \cdot T(n-4) + 8c + 4c + 2c + c.$$

$$\therefore T(n) = c + 2c + 4c + 8c + \cdots + 2^{n-1} \cdot \underbrace{T(n-(n-1))}_{T(1) = c}$$

$$= c \left[1 + 2 + 4 + 8 + \cdots + 2^{n-1} \right]$$

$$= c \cdot \left(\frac{2^n - 1}{2 - 1} \right) = c \cdot (2^n - 1)$$

$$\Rightarrow T(n) = \Theta(2^n).$$

Prob. 5 Koch Snowflake.

(5.1) Let K_n be a Koch snowflake of order n.

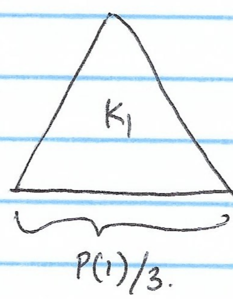
let $P(n)$ = Perimeter of K_n

$A(n)$ = Area under K_n.

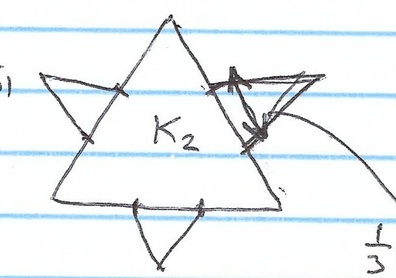
Claim: $P(n) = \frac{4}{3} \cdot P(n-1)$

Proof: We prove this by induction.

Base case: Consider K_1. This is an equilateral triangle with length of side being $P(1)/3$.



$K_2 \Rightarrow$ divide a side in K_1 into 3 equal parts and replace the middle part with 2 additional sides of same size



$\frac{1}{3}\left(\frac{P(1)}{3}\right) = \frac{P(1)}{3^2}$

$$\Rightarrow P(2) = 3\left[4 \cdot \frac{P(1)}{3^2}\right] = \frac{4}{3} P(1).$$

3 sides of K_1

4 sides of $\frac{P(1)}{3^2}$ length on each side of K_1.

P.T.O.

Intermediate case: let $P(i) = \frac{4}{3} P(i-1)$

$\forall\ i = 1, \ldots, m.$

Now, let us evaluate $P(m+1)$.

$K_m \Rightarrow 3 \cdot 4^{m-1}$ sides. $\Rightarrow$ each side has length $\dfrac{P(m)}{3 \cdot 4^{m-1}}$.

each side in K_m is replaced by 4 sides of length $\frac{1}{3} \cdot \left(\dfrac{P(m)}{3 \cdot 4^{m-1}} \right)$. to form K_{m+1}.

$$\Rightarrow \quad P(m+1) = 3 \cdot 4^{m-1} \left[4 \cdot \frac{1}{3} \cdot \frac{P(m)}{3 \cdot 4^{m-1}} \right]$$

$$= \frac{4}{3} \cdot P(m).$$

Terminal case:

for some terminal order $m = n$,

the recursion holds true as well.

$$\Rightarrow \quad P(n) = \frac{4}{3} \cdot P(n-1) \quad \square.$$

P.T.O.

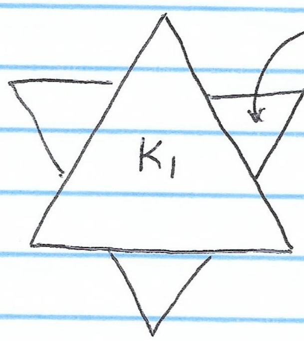
Claim: $A(n) = A(n-1) + \frac{1}{3}\cdot\left(\frac{4}{9}\right)^{n-1}\cdot A[1]$.

$\forall\ n = 2, 3, 4, \dots$

Proof:

Base case: Consider K_1. Then, $A(1) = \frac{\sqrt{3}}{4}\cdot s^2$.

where s is the side length of K_1.



an equilateral Δ^{le} of side length $\frac{s}{3}$.

$\Rightarrow$ its area $= \frac{\sqrt{3}}{4}\cdot\left(\frac{s}{3}\right)^2$.

$$\therefore\ A(2) = A(1) + 3\cdot\frac{\sqrt{3}}{4}\left(\frac{s}{3}\right)^2$$

$$= A(1) + \frac{1}{3}\cdot\frac{\sqrt{3}}{4}\cdot s^2$$

$$= A(1) + \frac{1}{3}\cdot\left(\frac{4}{9}\right)^0\cdot A(1)$$

Intermediate case:

Assume $A(i) = A(i-1) + \frac{1}{3}\left(\frac{4}{9}\right)^{i-2}\cdot A(1)$

$\forall\ i = 2, 3, \dots, m$.

Now, $A(m+1) = ?$

Note: K_m has $3\cdot 4^{m-1}$ sides, each of length $s/3^{m-1}$.

$$\therefore \quad A(m+1) = A(m) + 3 \cdot 4^{m-1}\left[\frac{\sqrt{3}}{4}\left(\frac{s}{3^{m-1}} \cdot \frac{1}{3}\right)^2\right]$$

$$= A(m) + \frac{1}{3} \cdot \left(\frac{4}{9}\right)^{m-1} \cdot \frac{\sqrt{3}}{4} \cdot s^2 \quad = A[1]$$

$$\Rightarrow \quad A(m+1) = A(m) + \frac{1}{3}\left(4/9\right)^{m-1} \cdot A(1).$$

Termination: If we are interested in K_n for some finite n, the recursion terminates and

$$A(n) = A(n-1) + \frac{1}{3} \cdot \left(\frac{4}{9}\right)^{n-2} \cdot A(1).$$

$\square$.

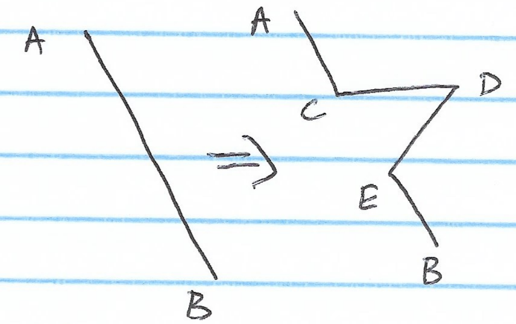
Prob. 5.2

```
Koch(n)
    V = Koch(n-1)
    for i = 1 to V.length
        if i ≤ V.length - 1
            U = alter_side(V[i], V[i+1])
            V = [V[1:i], U, V[i+1:n]]
        else
            U = alter_side(V[V.length], V[1])
            V = [V, U]
    return V.
```

The subroutine "alter_side()" is as follows:

P.T.O.

alter_side (A, B) → given the side AB, alter it into ACDE B



S = AB. length

find C s.t. $AC = S/3$

find E s.t. $BE = S/3$

find D s.t. CDE forms an equilateral Δ^{le} with side length $S/3$.

return C, D, E.

(5.3) **runtime**

$$T(n) = T(n-1) + \Theta(4^{n-1})$$

there are $3 \cdot 4^{n-1}$ vertices ⇒ the for loop in Koch(n) iterates so many times.

$$\therefore \quad T(n) = T(n-1) + c.4^{n-1}$$

$$= T(n-2) + c.\left[4^{n-2} + 4^{n-1}\right]$$

$$\vdots$$

$$= T(1) + c.\left[\sum_{i=1}^{n-1} 4^i\right] = T(1) + c\left(\frac{4^n - 1}{4 - 1}\right)$$

Assume $T(1) = \Theta(1)$, $T(n) = \Theta(4^n)$