Algorithms -- COMP.4040 Honor Statement
(Courtesy of Prof. Tom Costello and Karen Daniels with modifications)

Must be attached to each submission

Academic achievement is ordinarily evaluated on the basis of work that a student produces independently. Infringement of this Code of Honor entails penalties ranging from reprimand to suspension, dismissal or expulsion from the University.

Your name on any exercise is regarded as assurance and certification that what you are submitting for that exercise is the result of your own thoughts and study. Where collaboration is authorized, you should state very clearly which parts of any assignment were performed with collaboration and name your collaborators.

In writing examinations and quizzes, you are expected and required to respond entirely on the basis of your own memory and capacity, without any assistance whatsoever except such as what is specifically authorized by the instructor.

I certify that the work submitted with this assignment is mine and was generated in a manner consistent with this document, the course academic policy on the course website on Blackboard, and the UMass Lowell academic code.

Date: 06/08/2019

Name (please print): PHONG VO

Signature: Phong

Due Date: June 6 (M), BEFORE the class begins

This assignment covers textbook 7.1, 7.2 and Chapter1~4.

1. **QuickSort Algorithm (10 points)**
Exercise 7.1-1 (p173)

2. QuickSort Algorithm (25 points)
Using the PARTITION and QUICKSORT routines in textbook 171, what value of q does each PARTITION return, when all elements in A[1..n] are distinct and sorted in *descending* order? Justify your answer.

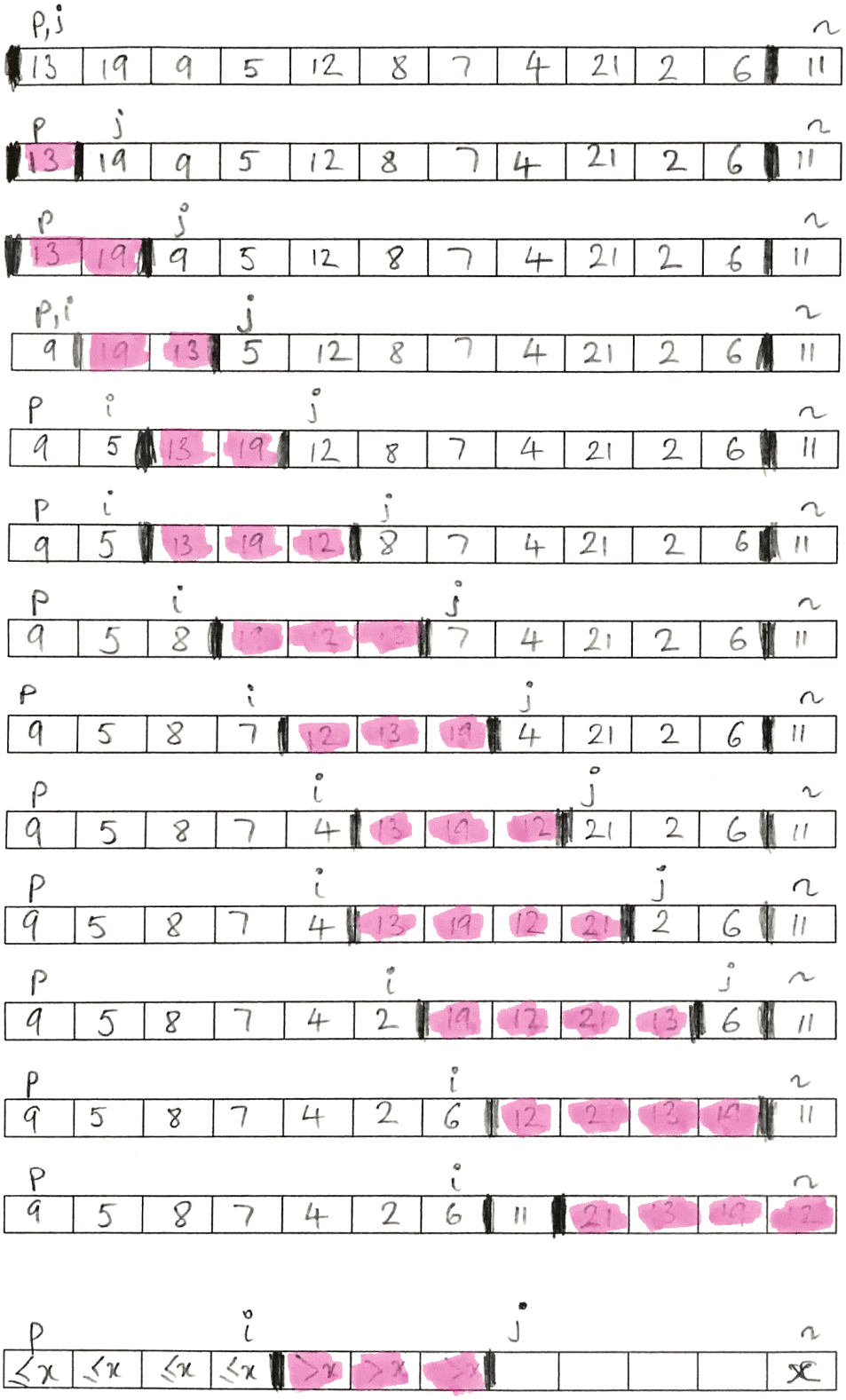
3. **QuickSort Algorithm Running Time (20 points)**
Derive a recurrence for the running time of the QUICKSORT algorithm (p171) for the above case in Problem 2. Solve the recurrence by providing tight upper and lower bounds on the running time. Justify your answer.

4. **QuickSort and Substitution (25 points)**
Use the substitution method to prove your answer in 3.

5. **QuickSort Analysis (20 points)**
Assume the partitioning algorithm always produces an 80-to-20 proportional split, write the recurrence of the running time of QuickSort in this case. Solve the recurrence by using a recursion tree.

Problem 1: Exercise 7.1-1 (p 173)

Array A = [13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11]



HW 4

(2)

Suppose we have an array, which is already descending sorted

$A = \langle 5, 4, 3, 2, 1 \rangle$

$q = r$, is always returned with r is the last index of each subarray. either first or last index

-5.

QUICKSORT$(A, 1, 5)$ calls PARTITION$(A, 1, 5)$

p and r are swapped $\Rightarrow$ returns $\boxed{q = 1}$

QUICKSORT$(A, q+1, r)$ is equivalent with QUICKSORT$(A, 2, 5)$

calls PARTITION$(A, 2, 5)$ $\Rightarrow$ returns $\boxed{q = 5}$

QUICKSORT$(A, p, q-1)$ is equivalent with QUICKSORT$(A, 2, 4)$

p and r are swapped $\Rightarrow$ returns $\boxed{q = 2}$

QUICKSORT$(A, q+1, r)$ is equivalent with QUICKSORT$(A, 3, 4)$

calls PARTITION$(A, 3, 4)$ $\Rightarrow$ returns $\boxed{q = 4}$

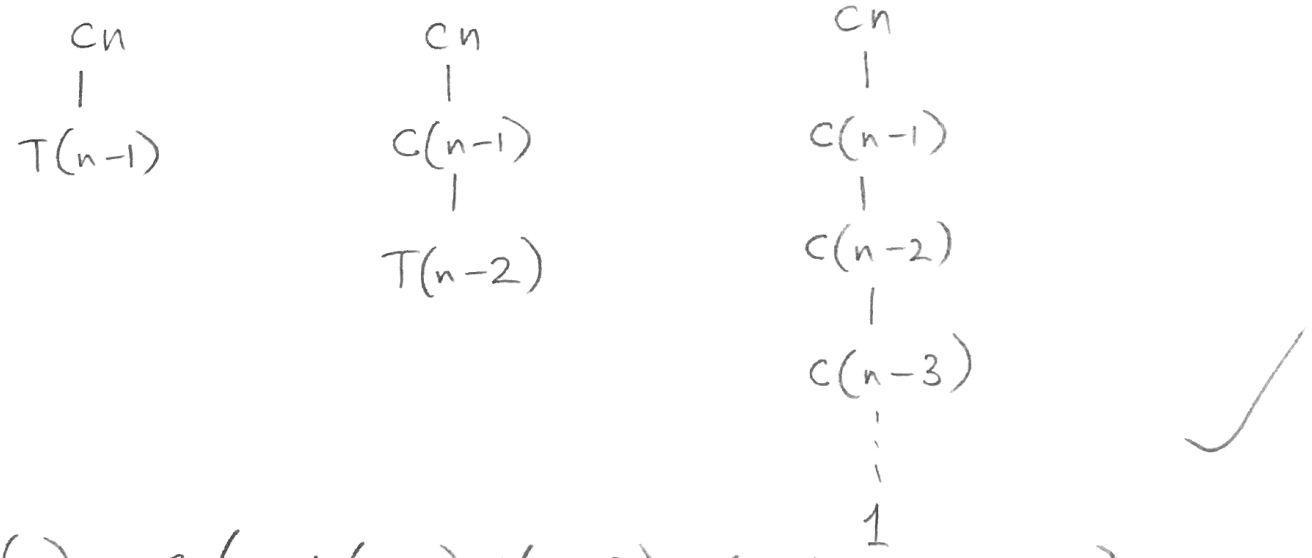
(3) Quick Sort algorithm running time

Quick Sort (A, p, r)	cost	Numbers of times
1. if $p < r$	c_1	1
2. q = PARTITION (A, p, r)	$\theta(n)$	1
3. QUICKSORT (A, p, $q-1$)	$T(0)$	1
4. QUICK SORT (A, $q+1$, r)	$T(n-1)$	1

$$T(n) = c_1 + \theta(n) + T(0) + T(n-1)$$

$$T(n) = T(n-1) + \theta(n) = T(n-1) + cn$$

$$T(n-1) = T(n-2) + c(n-1)$$



$$T(n) = c\left(n + (n-1) + (n-2) + (n-3) + \dots + 1\right)$$

$$= c\,\frac{(n+1)n}{2} = c\,\frac{n^2+n}{2} = \boxed{\theta(n^2)}$$

④

$T(n) = T(n-1) + cn$ (c: positive const.)

* Upper bound: $T(n) \leq T(n-1) + cn$

Guess $T(n) = O(n^2)$

$T(n) \leq dn^2$ (d: pos. const.)

$T(n-1) \leq d(n-1)^2$

$= d(n^2 - 2n + 1)$

Substitution: $T(n) \leq d(n^2 - 2n + 1) + cn$

$= dn^2 - n(2d - c) + d$

$\leq dn^2$ if $-n(2d-c) + d \leq 0$

$\Rightarrow 2d - c \geq 0 \Rightarrow c \leq 2d$

Hence: upper bound of $T(n) = O(n^2)$

* Lower bound: $T(n) \geq T(n-1) + cn$ (c: pos. const.)

Guess $T(n) = \Omega(n^2)$

$\geq dn^2$ (d: pos. const.)

$T(n-1) \geq d(n-1)^2$

Substitution: $T(n) \geq d(n-1)^2 + cn$

$= d(n^2 - 2n + 1) + cn$

$= dn^2 + (c - 2d)n + d$

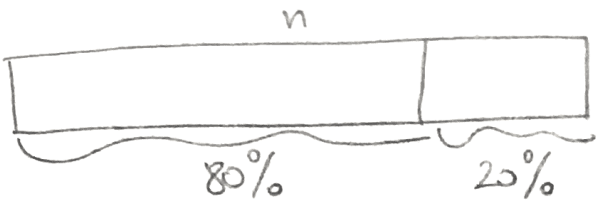
$\geq dn^2$ if $(c - 2d)n + d \geq 0$

$\Rightarrow c - 2d \geq 0 \Rightarrow c \geq 2d$

Hence, lower bound of $T(n) = \Omega(n^2)$

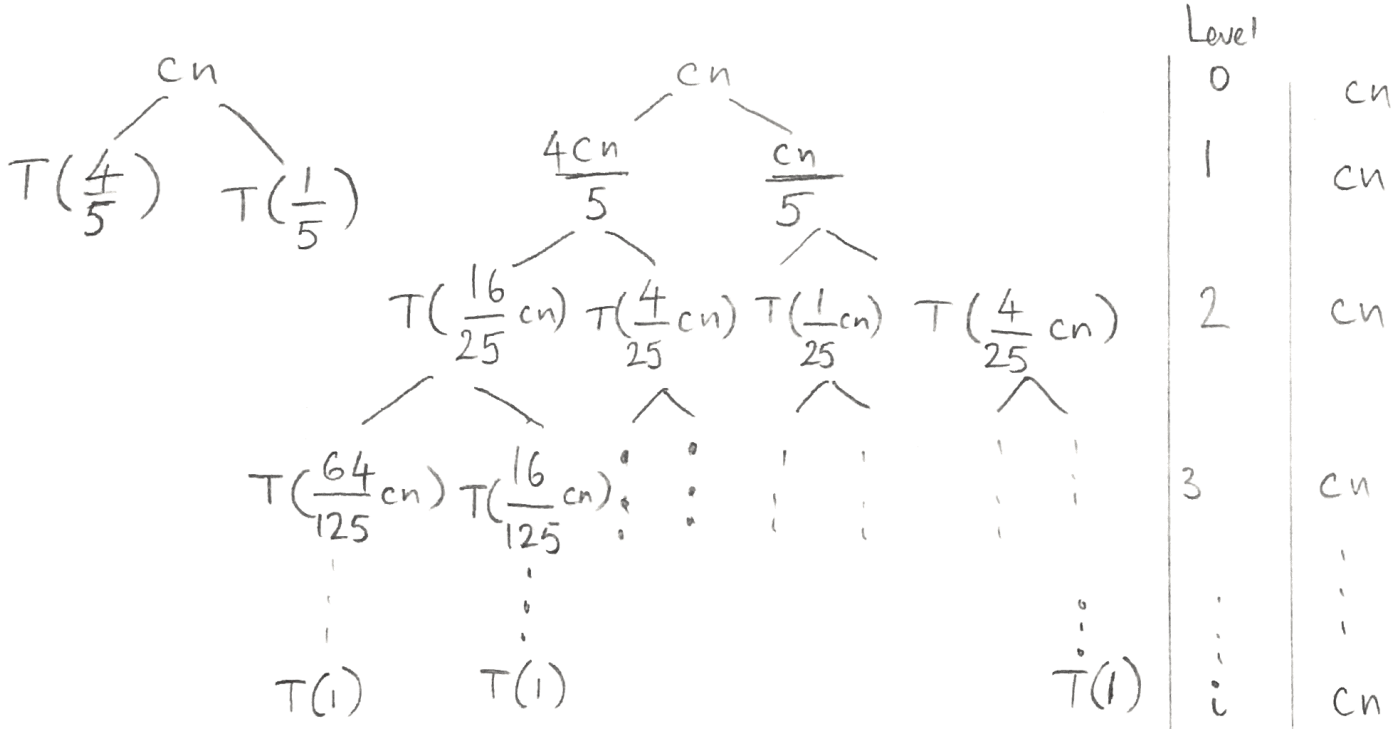
We have $T(n) = O(n^2)$ and $T(n) = \Omega(n^2)$, so $\boxed{T(n) = \Theta(n^2)}$

⑤



$T(n) = T(\frac{80}{100}n) + T(\frac{20}{100}n) + cn$ (c: pos. const.)

$= T(\frac{4}{5}n) + T(\frac{1}{5}n) + cn$

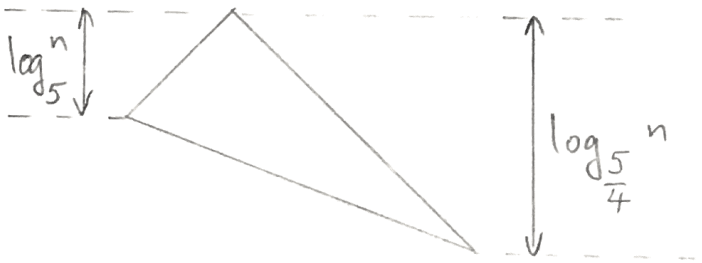


At level i: . the left most leaf $= \frac{4^i n}{5^i} = 1 \Leftrightarrow (\frac{4}{5})^i n = 1$

$\Rightarrow n = (\frac{5}{4})^i \Rightarrow i = \log_{\frac{5}{4}} n$

. the right most leaf $= \frac{1^i}{5^i} n = 1 \Rightarrow n = 5^i$

$\Rightarrow i = \log_5 n$



$T(n) \geq \sum_{i=0}^{\log_5 n} cn = \Omega(n \lg n)$

$T(n) \leq \sum_{i=0}^{\log_{5/4} n} cn = O(n \lg n)$

Therefore, $T(n) = \Theta(n \lg n)$ ✓ 95/100