COMP.4040 HW4 Solution

1. **Solution**(credit from Salvati Douglas):

a. When all elements in $A[p \dots r]$ are distinct and sorted in *descending* order:

In an array $A = [5, 4, 3, 2, 1]$, q always returns either $q = r, r$ being the last element of the subarray.

5 | 4 | 3 | 2 | 1	QuickSort(A,**1**,**5**) Calls Partition(A,**1**,**5**) Partition returns $q = 1$ Switch $A[p]$ with $A[r]$
1 | 4 | 3 | 2 | 5	QuickSort(A,q+**1**,r) QuickSort(A,**2**,**5**) Calls Partition(A,**2**,**5**) Partition returns $q = 5$
1 | 4 | 3 | 2 | 5	QuickSort(A,p,q-**1**) Quicksort(A,**2**,**4**) Calls Partition(A,**2**,**4**) Partition returns $q = 2$ Switch $A[p]$ and $A[r]$
1 | 2 | 3 | 4 | 5	QuickSort(A,q+**1**,r) QuickSort(A,**3**,**4**) Calls Partition(A,3,**4**) Partition returns $q = 4$

b. When all elements in $A[p \ldots r]$ are distinct and sorted in *ascending* order:

In an array $A = [1,2,3,4,5]$, q always returns $q = r$, r being the last element in each subarray.

1 | 2 | 3 | 4 | 5	QuickSort(A,**1**,**5**) Calls Partition(A,**1**,**5**) Partition returns $q = 5$
1 | 2 | 3 | 4 | 5	QuickSort(A,p,q-**1**) QuickSort(A,**1**,**4**) Calls Partition(A,**1**,**4**) Partition returns $q = 4$
1 | 2 | 3 | 4 | 5	QuickSort(A,p,q-**1**) Quicksort(A,**1**,**3**) Calls Partition(A,**1**,**3**) Partition returns $q = 3$
1 | 2 | 3 | 4 | 5	QuickSort(A,p,q-**1**) Quicksort(A,**1**,**2**) Calls Partition(A,**1**,**2**) Partition returns $q = 2$

2. Solution(credit from Gao Gao):

2. The running time of Partition is $f(n) = \Theta(n)$. In the problems above, since one of the two sub arrays in QS is always 0, the running time of QS is $T(n) = T(n-1) + T(0) + \Theta(n)$

$$= T(n-1) + \Theta(n)$$

Using recursion trees

$$\begin{array}{ccc} cn & cn & cn \\ | & | & | \\ T(n-1) & c(n-1) & c(n-1) \\ & | & | \\ & T(n-2) & c(n-2) \\ & & \vdots \\ & & T(1) \end{array}$$

$$T(n) = c\left(n + (n-1) + (n-2) + \dots + 1\right)$$
$$= c\,\frac{n(n-1)}{2}$$
$$= \Theta(n^2)$$

A tight bound for the problems above is $T(n) = \Theta(n^2)$

3. $T(n) = T(n-1) + \Theta(n)$

Upper Bound:

Guess: $T(n) = O(n^2)$, so $T(n) \le cn^2 + dn$, where $c, d > 0$ and are constant

Substitute: $T(n) \le c(n-1)^2 + \Theta(n)$

$$= c(n-1)^2 + dn$$
$$= c(n-1)(n-1) + dn$$
$$= c(n^2 - 2n + 1) + dn$$
$$= cn^2 - 2cn + c + dn$$
$$= cn^2 - (2n-1)c + dn$$

For $-(2n-1)c + dn < 0$, pick c, d so statement is true. $d > \frac{c}{2}$

$\therefore T(n) \le cn^2 \Rightarrow T(n) = O(n^2)$

Lower Bound:

Guess: $T(n) = \Omega(n^2)$, so $T(n) \ge cn^2 + dn$, where $c, d > 0$ and constant

Substitute: $T(n) \ge c(n-1)^2 + \Theta(n)$

$$= c(n-1)(n-1) + dn$$
$$= c(n^2 - 2n + 1) + dn$$
$$= cn^2 - 2cn + c + dn$$

$= cn^2 - (2n-1)c + dn$, choose c, d to make $d \le \frac{c}{2}$

$\therefore T(n) \ge cn^2 \Rightarrow T(n) = \Omega(n^2)$

so $T(n) = \Theta(n^2)$

3. Solution(credit from Evan O'Leary):

Upper-bound : $T(n) = O(n^2)$

Guess: $T(n) = O(n^2)$, so $T(n) \leq cn^2 + dn$, where c, d are positive constants

Substitution: $T(n) \leq c(n-1)^2 + \theta(n)$

$$
\begin{aligned}
T(n) &\leq c(n-1)^2 + dn \\
&= c(n-1)(n-1) + dn \\
&= c(n^2 - 2n + 1) + dn \\
&= cn^2 - 2cn + c + dn \\
&= cn^2 - (2n-1)c + dn
\end{aligned}
$$

$\therefore$ For $-(2n-1)c + dn < 0$, a c, d need to be picked so that this is true

This simplifies down to $d > \frac{cn}{2n-1}$, thus we can conclude $d > \frac{c}{2}$.

Thus,

$T(n) \leq cn^2 \Rightarrow T(n) = O(n^2)$

Lower-bound : $T(n) = \Omega(n^2)$

Guess: $T(n) = \Omega(n^2)$, so $T(n) \geq cn^2 + dn$, where c, d are positive constants.

Substitution: $T(n) \geq c(n-1)^2 + \theta(n)$

$$
\begin{aligned}
T(n) &\geq c(n-1)^2 + dn \\
&= c(n-1)(n-1) + dn \\
&= c(n^2 - 2n + 1) + dn \\
&= cn^2 - 2cn + c + dn \\
&= cn^2 - (2n-1)c + dn
\end{aligned}
$$

$\therefore$ For $-(2n-1)c + \theta(n) \geq 0$, we need to pick c, d so that $0 < d \leq \frac{c}{2}$

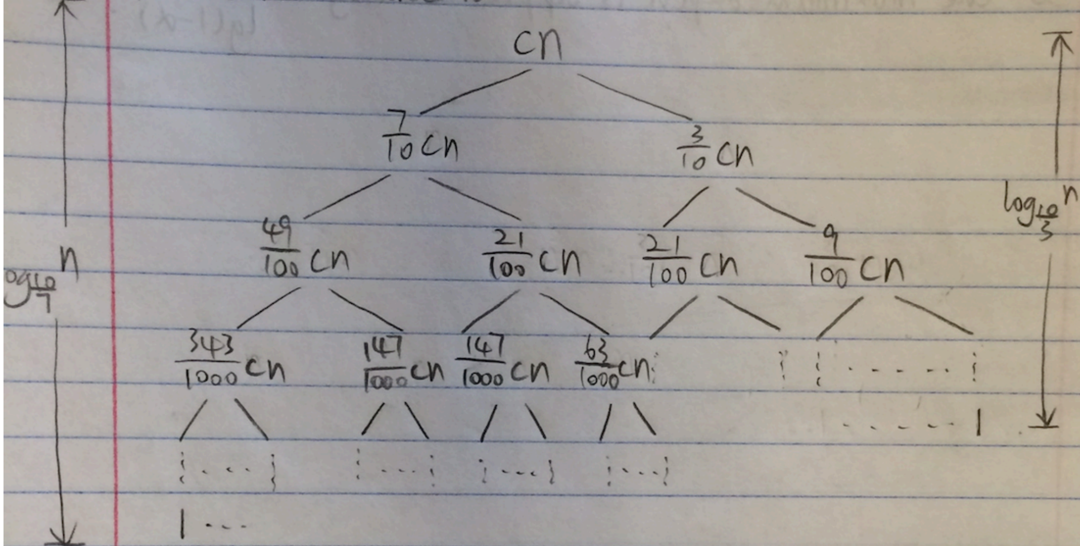
Thus,

$T(n) \geq cn^2 \Rightarrow T(n) = \Omega(n^2)$

$\boxed{\text{Therefore, } T(n) = \theta(n^2)}$

4. Solution(credit from Wei Ma):

The recursion tree is:



Leftmost branch peters out after $\log_{\frac{10}{7}} n$ levels

Rightmost branch peters out after $\log_{\frac{10}{3}} n$ levels

Therefore, there are $\log_{\frac{10}{3}} n$ full levels, the cost for each level is cn. There are $\log_{\frac{10}{7}} n$ levels, the cost for each level is <u>less than</u> cn.

The lower bound guess: $T(n) \geq c \cdot n \log_{\frac{10}{3}} n = \Omega(n \lg n)$.

The upper bound guess: $T(n) \leq c \cdot n \log_{\frac{10}{7}} n = O(n \lg n)$.

Therefore: $T(n) = \Theta(n \lg n)$.

5. Solution(credit from Gao Gao):

QuickSort Analysis

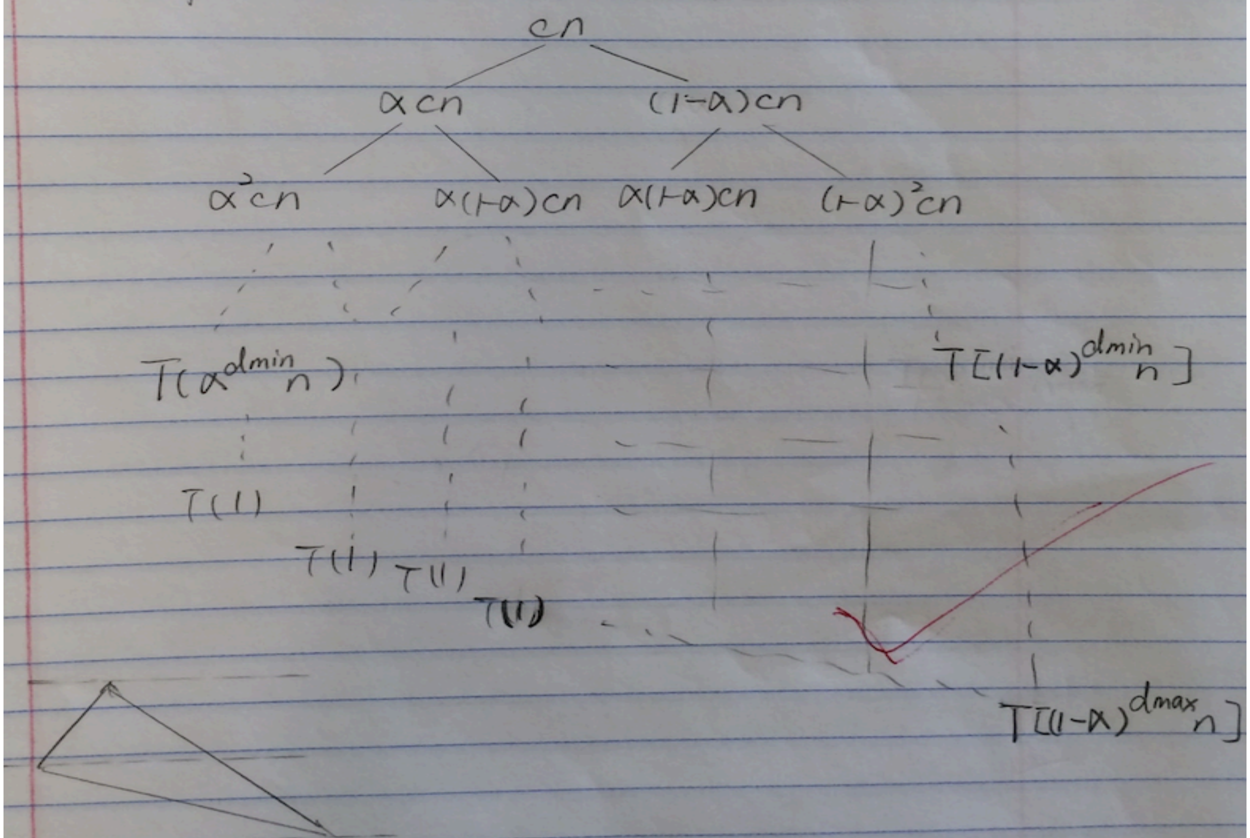
Because the splits at every level of QuickSort are in the proportion $1-\alpha$ to α. $0 < \alpha \le \frac{1}{2}$ So $1-\alpha > \alpha$

$$d_{min} = \frac{-\lg n}{\lg \alpha} \qquad d_{max} = \frac{-\lg n}{\lg(1-\alpha)}$$

So By the QuickSort, the recurrence for the running time is

$$T(n) = T(\alpha n) + T[(1-\alpha)n] + \Theta(n)$$

we found the recurrence tree is



Assume the minimum depth of a leaf in the recursion tree

$1-\alpha > \alpha$. $\quad \alpha^{d_{min}} \cdot n = 1$

there fore $\quad d_{min} = \log_\alpha \frac{1}{n} = \frac{\lg \frac{1}{n}}{\lg \alpha} = -\frac{\lg n}{\lg \alpha}$

Assume the maximum depth of a leaf in the recursion tree is d_{max}

$$1-\alpha > \alpha$$

$$(1-\alpha)^{d_{max}} \cdot n = 1$$

So $$d_{max} = \log_{(1-\alpha)} \frac{1}{n} = -\frac{\lg n}{\lg(1-\alpha)}$$