Name: Inchara Raveendra

SCU ID: 00001653600

Assignment - 4

8.1-1
What is the smallest possible depth of a leaf in a decision tree for a comparison sort?

For an input sequence of 'n' elements, it requires n-1 comparisons. Thus, a decision tree would have a least depth of at least n-1

8.1-3
Show that there is no comparison sort whose running time is linear for at least half of the $n!$ inputs of length n. What about a fraction of $1/n$ of the inputs of length n? What about a fraction $1/2^n$?

$h \rightarrow$ Height of the decision tree

$n! \rightarrow$ Total number of possible input permutations

a) Half of the inputs

$$h \geq \log_2 \left(\frac{n!}{2}\right)$$

$$= \log_2 n! - \log_2 2$$

$$\geq n\log_2 n - n\log_2 e - 1$$

$$= \Omega(n\log_2 n)$$

b) Fraction of $1/n$

$$h \geq \log_2\left(\frac{n!}{n}\right)$$

$$= \log_2 n! - \log_2 n$$

$$\geq n\log_2 n - n\log_2 e - \log_2 n$$

$$= \Omega(n\log_2 n)$$

c) Fraction of $1/2^n$

$$h \geq \log_2\left(\frac{n!}{2^n}\right)$$

$$= \log_2 n! - n$$

$$\geq n\log_2 n - n\log_2 e - n$$

$$= \Omega(n\log_2 n)$$

$\therefore$ In all the above cases, there is no comparison sort with a linear running time

8.1-4

Suppose that you are given a sequence of n elements to sort. The input sequence consists of n/k subsequences, each containing k elements. The elements in a given subsequence are all smaller than the elements in the succeeding subsequence and larger than the elements in the preceding subsequence. Thus, all that is needed to sort the whole sequence of length n is to sort the k elements in each of the n/k subsequences. Show an $\Omega(n \lg k)$ lower bound on the number of comparisons needed to solve this variant of the sorting problem. (*Hint:* It is not rigorous to simply combine the lower bounds for the individual subsequences.)

Let A be a sequence of 'n' elements

For any comparison sort, consider a decision tree of height 'h'

No. of subsequences $= \frac{n}{k}$

No. of input permutations $= (k!)^{n/k}$ = Number of leaves

Since, $2^h \geq (k!)^{n/k}$

Take $\log_2$ on both sides

$$h \geq \frac{n}{k} \log_2 (k!)$$

$$\geq \frac{n}{\cancel{k}} \times \frac{\cancel{k}}{2} \log_2 (k/2)$$

$$\geq (n/2) \log_2 (k/2)$$

$\therefore$ worst-case running time of any comparison-based sorting algorithm is $\Omega((n/2)\log_2(k/2)) = \Omega(n \log_2 k)$

8.3-1

Using Figure 8.3 as a model, illustrate the operation of RADIX-SORT on the following list of English words: COW, DOG, SEA, RUG, ROW, MOB, BOX, TAB, BAR, EAR, TAR, DIG, BIG, TEA, NOW, FOX.

Radix sort uses counting sort as a sub-routine and sorts the least-significant bit first. It then moves on to the second least significant bit. Until the most significant bit is sorted, this process continues

	$i=1$	$i=2$	$i=3$
COW	SEA	TAB	BAR
DOG	TEA	BAR	BIG
SEA	MOB	EAR	BOX
RUG	TAB	TAR	COW
ROW	DOG	SEA	DIG
MOB	RUG	TEA	DOG
BOX	DIG	DIG	EAR

TAB	→	BIG	→	BIG	→	FOX
BAR		BAR		MOB		MOB
EAR		EAR		DOG		NOW
TAR		TAR		COW		ROW
DIG		COW		ROW		RUG
BIG		ROW		NOW		SEA
TEA		NOW		BOX		TAB
NOW		BOX		FOX		TAR
FOX		FOX		RUG		TEA

8-6 *Lower bound on merging sorted lists*

The problem of merging two sorted lists arises frequently. We have seen a procedure for it as the subroutine MERGE in Section 2.3.1. In this problem, we will prove a lower bound of $2n - 1$ on the worst-case number of comparisons required to merge two sorted lists, each containing n items.

First we will show a lower bound of $2n - o(n)$ comparisons by using a decision tree.

a. Given $2n$ numbers, compute the number of possible ways to divide them into two sorted lists, each with n numbers.

b. Using a decision tree and your answer to part (a), show that any algorithm that correctly merges two sorted lists must perform at least $2n - o(n)$ comparisons.

Now we will show a slightly tighter $2n - 1$ bound.

c. Show that if two elements are consecutive in the sorted order and from different lists, then they must be compared.

d. Use your answer to the previous part to show a lower bound of $2n - 1$ comparisons for merging two sorted lists.

a) There are $\binom{2n}{n}$ ways to divide $2n$ numbers into 2 sorted lists, each with 'n' numbers.

b) $$\binom{2n}{n} < 2^n$$

$$\binom{2n}{n} \approx \frac{2n!}{(n!)^2}$$

Using Stirling's approximation,

$$\sqrt{4\pi n}\,(2n/e)^2 / (\sqrt{2\pi n}\, e^{1/12n})^2$$

$$2^n = 2n - \log 2\pi n$$
$$2^n = 2n - O(n)$$

c) Since the elements are from different lists, to know the relationship we have to compare them and also know the order to arrange in the final array.

d) Let L be the first sorted list
$L = l_1, l_2, \ldots, l_n$ and second sorted list
$R = r_1, r_2, \ldots, r_n$

If all elements in L are equal and greater than every element in R then, merge procedure will compare l_1 with every element in L, no. of comparisons $= n-1$

According to c) we also compare if 2 elements are consecutive in sorted order and from a different list. Since $l_1 > r_1, r_2, \ldots, r_n$, no. of comparisons $= n$

$\therefore$ Running time $= n-1+n = 2n-1$