南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Problem 0 [50 points]

Student ID: 12312710 Student Name: 唐炽智.

Lab: ☐ Tue.5-6 Tang ☐ Tue.5-6 Wang ☐ Tue.5-6 Shen

☐ Wed.3-4 Wang ☐ Wed.3-4 Shen ☐ Wed.5-6 Wang ☑ Wed.5-6 Shen

Problem 1 [20 points] Heap Building Time Complexity Proof

The time complexity of turn sized-n array A into a binary heap on S via root-fix operator on dynamic array is O(n), where A stores the values in set S.

20

Suppose the array size is n. n is an positive integer

① then there's ~~$O(\frac{n}{2})$~~ about $\frac{n}{2}$ of leaves. which requires 0 comparison.

② for the nodes that have 1 or 2 children. it's in the ~~top~~ index $(\frac{n}{4}, \frac{n}{2})$ and those needs 1 comparison to determine whether it needs swap.

③ for those $(\frac{n}{8}, \frac{n}{4})$. 2 comparison is needed.

....

⓱ for the root, $\log_2 n = h$ comparison is needed.

in Summary: the whole algorizm takes $\frac{n}{4}\cdot 1 + \frac{n}{8}\cdot 2 + \frac{n}{16}\cdot 3 + \cdots \frac{n}{2^{h+1}}\cdot h$

times of comparison

$n\left(\sum_{i=1}^{h} \frac{i}{2^{i+1}}\right)$ we did calculation that $\sum_{i=1}^{h} \frac{i}{2^{i+1}} = 1 - \frac{1}{2^{(h+1)}}(h+2)$

When $h \to \infty$, $\sum_{i=1}^{h} \frac{i}{2^{i+1}} = 1$, is a constant.

∴ the algorizm's time complexity is $O(n)$

41+40

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Problem 2 [30 points] Huffman Encoding

Given (character, frequency) pairs as following:

H	N	S	O	E	Y	T	D
14	16	8	12	30	18	43	65

30

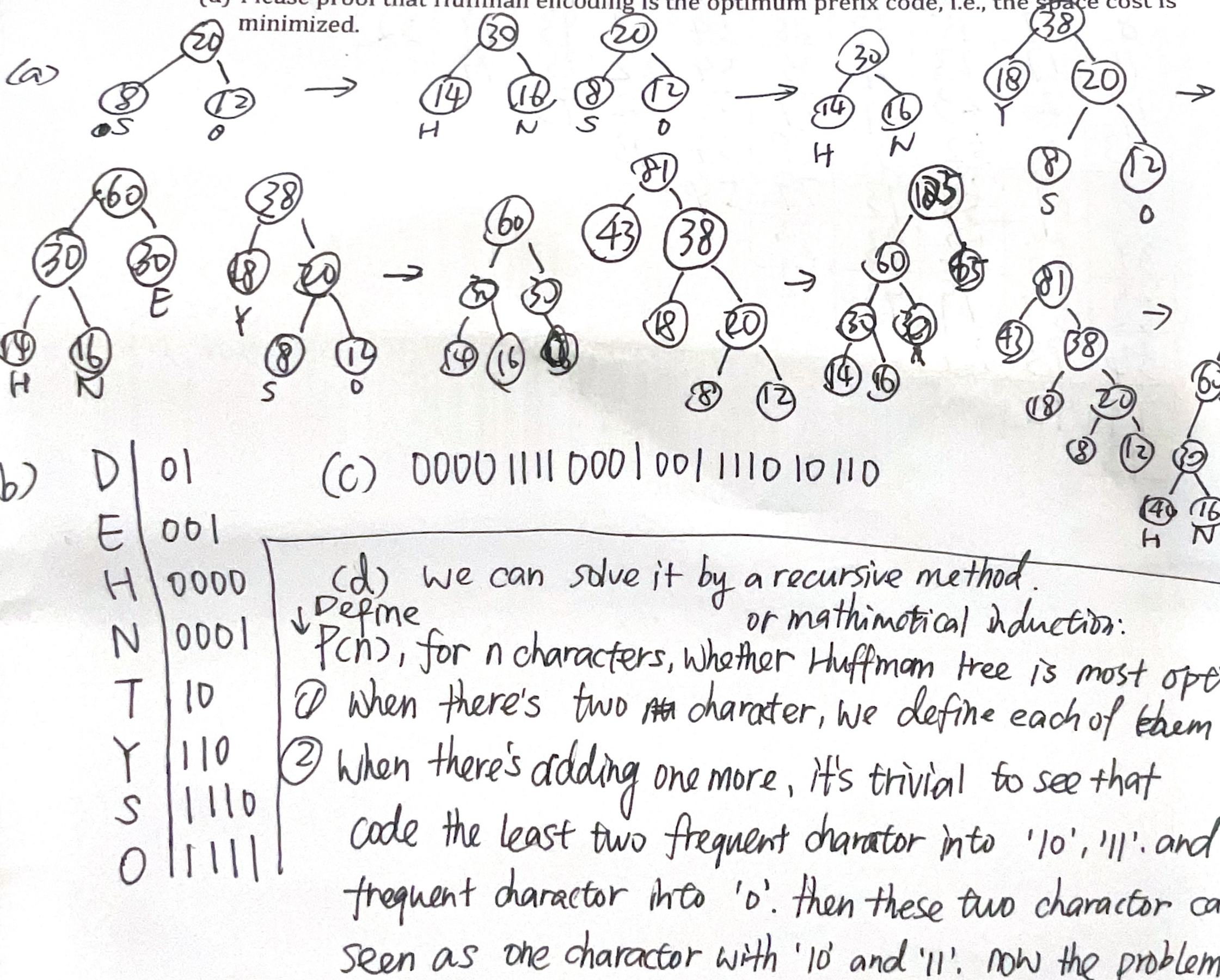
(a) Show the detail steps of building its Huffman tree, i.e., draw the Huffman tree building process step by step

(b) Write down the corresponding scheme of the Huffman tree you obtained in (a), you only need draw a table, which contains two columns, the left is the character, the right is its corresponding Huffman coding

(c) Write down the corresponding codes of string "HONESTY"

(d) Please proof that Huffman encoding is the optimum prefix code, i.e., the space cost is minimized.

(a)



(b)

D	01
E	001
H	0000
N	0001
T	10
Y	110
S	1110
O	1111

(c) 0000 1111 000 1001 1110 10110

(d) we can solve it by a recursive method.
or mathimotical induction:

↓Define
P(n), for n characters, whether Huffman tree is most optimum.

① When there's two ~~th~~ character, we define each of them '0', '1'

② When there's adding one more, it's trivial to see that code the least two frequent character into '10', '11'. and mostly frequent character into '0'. then these two character can be seen as one character with '10' and '11'. now the problem is turning into ① cases.

∴ we prove that for n≥2, P(2) is true, P(n)→P(n+1)

∴ by mathmatical induction, we proof that Huffman tree is always the most optimum code for all integer n≥2.

20

Problem 3 [20 points] Heap sort: a sorted array is created by repeatedly removing the root of the min-heap until the min-heap becomes empty. Given an array-based min-heap (as follows), fulfill the following table to show the elements in the min-heap during heap sort progress.

1	6	15	17	8	54	23	93	39	52	26	79
6	8	15	17	26	54	23	93	39	52	79	
8	17	15	39	26	54	23	93	79	52		
15	17	23	39	26	54	52	93	79			
17	26	23	39	79	54	52	93				
23	26	52	39	79	54	93					
26	39	52	93	79	54						
39	54	52	93	79							
52	54	79	93								
54	93	79									
79	93										
93											