/*

* Name: Huang Chao, Yeung-Kit Wong

* Date: Aug 09, 2016

* File: Checkpoint.pdf

*/

# PA2 Checkpoint PDF

**1. Describe how you built the tree.**

i) Calculate the frequencies of each character in the original file, then save into a vector, using ascii value of each character as index and count as value of each element.

ii) Use this vector to build the HCTree.

iii) Check if the vector is empty

iv) Save this vector to a priority queue

v) If the priority queue only has one node, add a dummy node

vi) Pull top two nodes from the queue and connect to a new dummy node (parent node), and the count of this dummy node is the sum of the count of these two nodes. Put dummy node back in queue and repeat step vi) until one node left in the queue.

vii) The last node is the root of the HCTree.

**2. Describe how you find the codeword for each byte.**

i) Create a vector to store the codeword

ii) Find the leaf node containing a specific symbol

iii) Traverse from that leaf node to root

iv) Keep track of current node

v) Check if current node is at child0 or child1

vi) Put the corresponding 0 or 1 into the vector

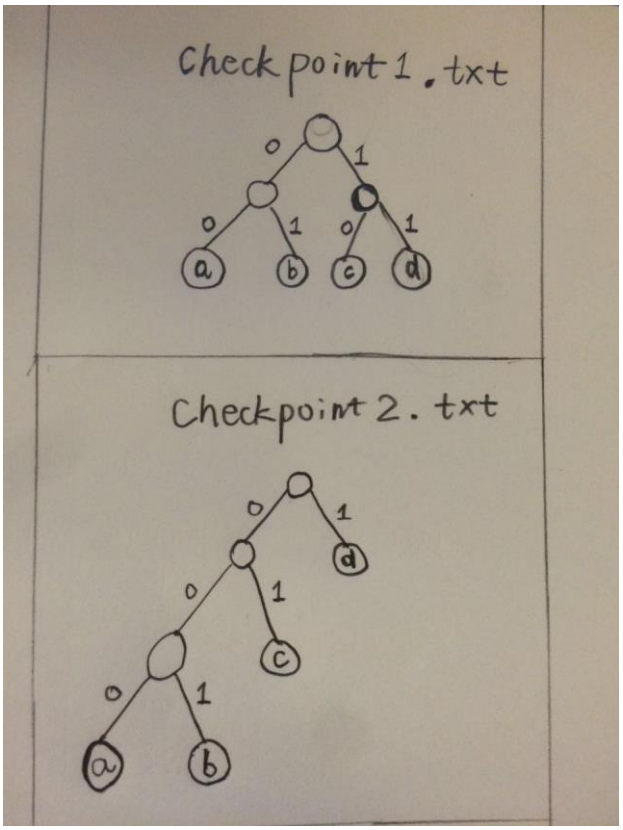
vii) Move current node to its parent node, repeat step v) until it reaches root node

viii) Read the number in the vector backward to get the codeword representing the symbol.

**3. Draw the Huffman Tree by hand and include the result in the writeup.**

| Checkpoint 1.txt | | |
|---|---|---|
| **Symbol** | **Count** | **Codeword** |
| a | 10 | 00 |
| b | 10 | 01 |
| c | 10 | 10 |
| d | 10 | 11 |

| Checkpoint 2.txt | | |
|---|---|---|
| **Symbol** | **Count** | **Codeword** |
| a | 4 | 000 |
| b | 8 | 001 |
| c | 16 | 01 |
| d | 32 | 1 |




**4. Explain why your handcoded text is different from your compressor output.**

**Answer:** Our handcoded text is the same as the compressor output for both checkpoint1.txt and checkpoint2.txt file.

5. Explain how you fixed it.

**Answer:** They have no difference, no fix needed.