

Data Structure - Homework3

Introduction

Huffman Coding在資料壓縮領域是相當重要的演算法，演算過程需從資料組中取出最小的兩值進行運算，因此適合使用資料結構中的優先佇列(Priority Queue)來實作，本次作業將要求使用Min-Heap來建構Huffman-Tree並完成編碼工作，同時練習有關樹的基本操作。

<https://zh.wikipedia.org/zh-tw/霍夫曼编码> (<https://zh.wikipedia.org/zh-tw/%E9%9C%8D%E5%A4%AB%E6%9B%BC%E7%BC%96%E7%A0%81>)

Problem

1. (20%)請實作Min-Heap，把輸入的符號與權重序列先建成節點(Leaf Node)後Push到Min-Heap中，以樹的形式將Min-Heap所存節點之權重值與樹高(從0起算)印出，範例如下

使用者需輸入size、符號序列及權重序列(以空白隔開)

```
Input size[int]: 9
Input symbol[char]: A D T S R U C E _
Input frequency[int]: 9 8 7 6 5 4 3 2 1
=====
MinHeap Tree:
1__
2__4__
3__7__8__5__
9__6__
Max Level: 3
=====
```

```

// Example
// Huffman Tree Node
struct HuffNode {
    int data;
    char symbol;
    HuffNode *leftChild;
    HuffNode *rightChild;
};
// min-heap class
template <class T>
class MinHeap {
public:
    MinHeap(){ //initialize };
    bool IsEmpty();
    const T& Top();
    void Push(const T&);
    void Pop();
    void ChangeSize1D(T *a, const int oldSize, const int newSize);
    // ChangeSize1D 在課本第130頁
private:
    T *heap; // element array
    int heapSize; // number of elements in heap
    int capacity; // size of the array heap
};

int main() {
    int size;
    cin >> size;
    char *arr = new char[size];
    int *freq = new int [size];
    // Read_user_input();
    // while(cin >> x) you will need it;

    MinHeap<HuffNode> min_heap;
    // Initalize the leaf node
    for (int i = 0; i < size; i++) {
        HuffNode leaf_node;
        leaf_node.data = freq[i];
        leaf_node.symbol = arr[i];
        leaf_node.leftChild = leaf_node.rightChild = nullptr;
        min_heap.Push(leaf_node);
    }

    // print_tree(min_heap);
    // cout << complete_tree_max_level(min_heap) << endl;
    return 0;
}

```

評分

(10%) Complete-Tree Structure

(10%) Max Level(with complete tree)

2. (30%)請根據第1小題建立的Min-Heap，建立Huffman編碼樹，以前序、中序的方式從Root開始走訪並印出權值、樹高(從0起算)及節點個數，範例如下

演算法

- (1) 每次從Min-Heap中取出兩最小權值的節點(Pop兩次)
- (2) 將此兩節點建立成新節點後(權值相加且子點依[規定]擺放)，再Push回Min-Heap中
- (3) 直到Min-Heap內只剩一個節點(此為Huffman Tree的Root)

Ref : Huffman Coding (<https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/>)

```
=====
Huffman Tree:
Preorder: 45 27 15 8 7 12 6 3 2 1 3 6 18 9 5 4 9
Inorder: 8 15 7 27 2 3 1 6 3 12 6 45 5 9 4 18 9
Max Level: 5
Number of Node: 17
=====
```

[規定]: 左子值 > 右子值

評分

- (10%) Preorder & Inorder
- (10%) Max Level(with binary tree)
- (10%) Number of Node

3. (10%)請根據第2小題建立的Huffman編碼樹依[規定]對符號進行編碼，並印出編碼結果即可，範例如下

```
=====
Huffman Coding:
D : 111
T : 110
E : 10111
_ : 10110
C : 1010
S : 100
R : 011
U : 010
A : 00
=====
```

[規定]: 左枝編1，右枝編0

4. (10%)請根據第3小題的編碼表實作解碼工作，使用者可輸入一組0/1序列，對該組序列進行解碼來得到字詞，範例如下

```
=====
Input sequence for decode: 111001100010110100110011010101011001001110111
Decoded Huffman Data:
DATA_STRUCTURE
```

Ref : <https://www.geeksforgeeks.org/huffman-decoding/>
(<https://www.geeksforgeeks.org/huffman-decoding/>)

5. (30%)最後請修正第一題的輸入方式，使用者輸入一字串，求該字串中不重複的字元(symbol)與其個數(size)，及每個不重複字元出現的頻率(frequency)，範例如下

使用者只需輸入一字串即可

```
Input a string: AADDTSSSRUUCEE_
Your input size[int]: 9
Your input symbol[char]: A C D E R S T U _
Your input frequency[int]: 2 1 2 2 1 3 1 2 1
=====
```

因此最後程式輸出結果會為

```

Input a string: AADDTSSSRUUCEE_
Your input size[int]: 9
Your input symbol[char]: A C D E R S T U _
Your input frequency[int]: 2 1 2 2 1 3 1 2 1
=====
MinHeap Tree:
1__
1__1__
1__2__3__2__
2__2__
Max Level: 3

=====
Huffman Tree:
Preorder: 15 8 4 2 1 1 2 4 2 2 7 4 2 1 1 2 3
Inorder: 1 2 1 4 2 8 2 4 2 15 1 2 1 4 2 7 3
Max Level: 4
Number of Node: 17

=====
Huffman Coding:
T : 1111
_ : 1110
D : 110
E : 101
U : 100
R : 0111
C : 0110
A : 010
S : 00

=====
Input sequence for decode: 11001011111010111100011110111100011011111000111101

Decoded Huffman Data:
DATA_STRUCTURE

Process returned 0 (0x0)   execution time : 73.766 s
Press ENTER to continue.

```

Huffman Tree不唯一，只要每次任意取出最小兩值建樹即可

但據[規定]以及使用**Min-Heap**(如遇相同值，檢查是否需要調整)實作，所建立的Huffman Tree形式應相同，對字符的編碼也應相同。

- (1) 本次作業請依照**Min-Heap**的方式去實作Huffman Coding
- (2) **不要使用排序的方式實作**
- (3) **每題之間請用如圖上的"="區隔，以便助教批改**

Notices

作業規定

1. 除**stack, queue, vector, list, string**之外，禁止使用STL相關套件(**deque, map, set, ect...**)
2. 可使用**algorithm, math.h**等數學運算套件
3. 程式以**c++/c**語言為主
4. 請在程式碼中適時**加入註解**(未註解說明者將酌量扣分)
5. 可參考Reference撰寫，但**請勿完全抄襲Ref及同學作業**
6. **Deadline: 5/10(日) 23:55 moodle繳交**

繳交格式

1. 作業zip檔內須包含**程式原始碼(.cpp/.c)**、**執行檔(.exe)**及**pdf說明文件(readme.pdf)**

程式原始碼需包含所有小題，請勿分開寫在不同的Code
readme.pdf請說明程式碼如何運作

2. 檔名**HW3_學號_姓名_v1.zip**
v1為有實作第五題
v2為沒有實作第五題

Office Time

禮拜三、四下午15:00~17:00，請先寄信通知

Mail ds2020@dcmc.ee.ncku.edu.tw (<mailto:ds2020@dcmc.ee.ncku.edu.tw>)
