CMPSC 465 – Spring 2021 — Solutions to Homework 9

Josiah Kim, SID 948500821, `juk483`

April 1, 2021

1. Getting started

(a) I did not work in a group.

(b) I did not consult with any of my group members.

(c) I did not consult any non-class materials.

2. Heaviest Edge in a Cycle

Let A, B, C denote vertices of an undirected cycle in a graph G. The edge weights are as follows: $w_{(A,B)} = e * -2$, $w_{(B,C)} = e * -1$, $w_{(C,A)} = e*$. In this case, the edge connecting C and $A$is the heaviest edge. We know that a MST does not contain any cycles and that at least two of these edges must be in the MST since an MST contains $|V| - 1$ edges.

The MST can contain the conbination of these edges:
$w_{(C,A)} = e*$, $w_{(B,C)} = e * -1$
Total Cost: $2e * -1$
$w_{(C,A)} = e*$, $w_{(A,B)} = e * -2$
Total Cost: $2e * -2$
$w_{(A,B)} = e * -2$, $w_{(B,C)} = e * -1$
Total Cost: $2e * -3$

In the cases above, all the MST containing edge (C, A) has a lighter MST that does not have edge (C, A). Therefore, $e*$ cannot appear in any MST of G.

3. Huffman Encoding

(a) Possible frequencies that would yield the following code would be: $\{f_a, f_b, f_b + f_c, f_c\}$.

(b) This code cannot be obtained because 0 is the prefix 00 which means that a is in the path of c making it not a full binary tree.

(c) This code cannot be obtained because it will not produce a full binary tree. The node connecting to the right of the root node will only have one child, a.

4. Cost of a Prefix-Free Encoding

```
def find_encoding(f):
        T = initialize empty tree
        H = priority queue ordered by f
        W = priority queue ordered by c
        for i = 1 to n
        insert(H, i)
        insert(W, i)
        for k = n+1 to 2n-1
            i = extractmin(H)
            j = extractmin(H)
            n = extractmin(W)
            if f[i]+f[j] < f[i]+f[n]:
                    create node k in T with children i and j
                    f[k] = f[i] + f[j]
                    insert(H, k)
            if f[i]+f[j] > f[i]+f[n]:
                    create node k in T with children i and n
                    f[k] = f[i] + f[n]
                    insert(H, k)
```

By adding another extractmin() to the huffman encoding we are adding another $O(logm)$. Therefore, the new total running time will be: $O(m(logm)^2)$.