Advanced Data Structures(COP 5536)
Fall 2016
Programming Project Report

Akshat Bhardwaj
UFID-4352774
akshatbhardwaj@ufl.edu

PROJECT DESCRIPTION

The project requires to implement a system to find the most popular hashtags appeared from an input file containing one million hashtags. This was done using two data structures.

1. A Max Fibonacci Heap was implemented to keep a track of the frequencies of hashtags.
2. A Hash Table was used with the hashtag as its key and pointer to the corresponding node in the Fibonacci heap as its value.

STRUCTURE OF THE PROGRAM AND FUNCTION DESCRIPTIONS

There is one class that I have used to implement the programming assignment. The name of that class is ‘fHeap’. A structure named ‘fNode’ is used that has the following grouped list of variables under it - ‘key’, ‘value’, ‘fDegree’, ‘fParent’, ‘fChild’, ‘fLeft’, ‘fRight’, ‘cut’. The class contains 8 public functions namely ‘initializeFHeap’, ‘createFNode’, ‘insertFNode’, ‘removeMax’, ‘increaseKey’, ‘nodeCut’, ‘cascadeNodeCut’.

A brief description of the variables that are a part of the structure ‘fNode’ used:

Key - This is an string type variable which is used to store the string as key.

Value - This is an integer type variable which is used to store the frequency of the hashtags.

fDegree - This is an integer type variable which is used to store the degree of the resulting heap.

fParent - This is a type of struct fNode which is used as a pointer to the parent node.

fChild - This is a type of struct fNode which is used as a child pointer by every node that has a child.

fLeft - This is a type of struct fNode which is a pointer to the left sibling of a node.

fRight - This is a type of sruct fNode which is a pointer to the right sibling of a node.

Cut - This is a boolean type which has been used to indicate if a node has lost a child after being connected to its parent node.

The different public functions that have been used are:

initializeFHeap - This function is used for the purpose of initialization of the Fibonacci Heap.
Parameters - None
Return Type - fNode*

createFNode - This function is used to create a Node with a given value.
Parameters - int, the value of the created node.
Return Type - fNode*

insertFNode - This function is used is used to insert a node in the heap with the given value.
Parameters - fNode* , string, int, bool
Return Type - void

removeMax - This function is used to remove the maximum element from the heap.
Parameters - fNode*
Return Type - fNode*

Consolidate - This function is used to consolidate the Max heap after every maximum element has been removed from the heap.
Parameters - none
Return Type - int

increaseKey - This function is used to increase the frequency of a HashTag every time the same hash tag appears. It call the nodeCut and the cascadeNodeCut functions within itself.
Parameters - fNode*, int
Return Type - fNode*

nodeCut - This function is used to detach a node from its parent if the child becomes greater than the parent.
Parameters - fNode* , fNode* , fNode*
Return Type - int

cascadeNodeCut - This function is used to detach a node from its sibling list after a removeMax or increaseKey operation. The path from parent of the node to the root is followed and all the nodes with a cut value of True are cut from their sibling list and inserted into top level list. This is stopped at the first node with a cut value of false. For this node cut value is set to True.
Parameters - fNode* , fNode*
Return Type - int

fConnect - This function is used to perform the linking between the nodes so that the smaller root is the child of the bigger root.
Parameters - fNode* , fNode*
Return Type - int

The main function takes the input file as the input and performs the operations depending on the hashtag frequency. The result is then printed out.

CONCLUSION

The assignment has been successfully implemented using a Max Fibonacci Heap and a Hash Table has been successfully implemented within acceptable time limits for the given input files.