Advances Data Structures
(COP 5536) Fall 2018
Programming Project Report

Name: Anagha Sanjay Joshi

UFID-4515-4948

Email: anaghajoshi@ufl.edu

Project Description

The goal of this project is to find the most popular keywords for a search engine "Duck Duck Go". We are using Fibonacci heap data structure to find the maximum frequent word at any given time. In the scope of this project the input file is read from command line and then a Hash map is used to store the string and frequency. Fibonacci heaps is used to keep track of the frequencies and giving the maximum occurring word.

Basic workflow:

The keywordcounter class takes input from the input file and processes it line by line. It takes the string as key and frequency as value and maps it into a hashmap in java. If the key is new then it is inserted into the Fibonacci heap and if it is already present then the increasekey function is called. If the input line is just a number then removemax function is called that many times.

The removemax function returns the node with max value whose key and value are stored in arraylists. After removing the given amount of maximum occuring elements they are re inserted into the heap. The returned value from removemax is then printed to a output_file.txt file.

Working Environment:

Compiler:

Javac

Compiling and running instructions:

This project has been tested on thunder.cise.ufl.edu and Eclipse Luna. To execute the program you can remotely access thunder.cise.ufl.edu and use the following commands.

make- to compile the file

java keywordcounter "path of the input file"

enter path of the input file without the quotes.

Structure of the program and function prototypes

I have used 3 classes to implement this problem statement.

1. keywordcounter class: this class contains the main method and is used to take the input file, write the output to a text file and calling correct functions from heapops class as per the requirement in the input file.
2. node class: this class is used to create objects of node type.
3. heapops class: this class is used for performing main operations of the Fibonacci heap such as removemax, consolidate, increasekey etc.

Class keywordcounter:

Variable Name:	keyval
Datatype:	Hashmap<String,node>
Description:	Stores the keywords and the nodes in the Fibonacci heap.

Variable Name:	ops
Datatype:	Heapops
Description:	Create object of heapops class.

Variable Name:	st
Datatype:	String
Description:	String to read from input file line by line.

Variable Name:	pt
Datatype:	Pattern
Description:	Stores the pattern given. Used for regex later to match the format of input string. Used for $keyword frequency type input.

Variable Name:	p1
Datatype:	Pattern
Description:	Stores the pattern given. Used for regex later to match the format of input string. Used for removemax and only digit type of input.

Variable Name:	match
Datatype:	Matcher
Description:	Matches the input string for pattern in pt.

Variable Name:	match1
Datatype:	Matcher
Description:	Matches the input string for pattern in p1.

Variable Name:	old
Datatype:	Int
Description:	Stores the old value of the given keyword before increasekey is performed.

Variable Name:	output
Datatype:	ArrayList<String>
Description:	Stores the keys that are in key field of the output node returned by removemax.

Variable Name:	outputvals
Datatype:	ArrayList<Integer>
Description:	Stores the values that are in associated with the key field of the output node returned by removemax.

Variable Name:	q
Datatype:	node
Description:	Stores the node that is returned by removemax as the max node.

Class node:

Variable Name:	prev
Datatype:	node
Description:	Stores the pointer to the previous node in the linked list.

Variable Name:	next
Datatype:	node
Description:	Stores the pointer to the next node in the linked list.

Variable Name:	child
Datatype:	node
Description:	Stores the pointer to the child of the given node in the heap structure.

Variable Name:	parent
Datatype:	node
Description:	Stores the pointer to the parent of the given node in the heap structure.

Variable Name:	key
Datatype:	String
Description:	Stores the keyword associated with that node.

Variable Name:	value
Datatype:	int
Description:	Stores the frequency associated with that node.

Variable Name:	childcut
Datatype:	Boolean
Description:	Stores the childcut value of the node. False value means that no child has been cut from that node.

Variable Name:	degree
Datatype:	int
Description:	Stores the number of children the node has

Class heapops:

Variable Name:	maxnode
Datatype:	node
Description:	Stores the node that has maximum value in the heap structure.

Functions:

Function Name: public void insert(node node)		
Description	Inserts the node in the heap structure.	
Parameters	node	node that is to be inserted.
Return value	void	

Function Name: public void increasekey(node val,int key)		
Description	Increases the frequency value of the given node by key.	
Parameters	val key	node whose value is to be increased. the value by which the frequency is to be updated.
Return value	void	

Function Name: private void cut(node ptr,node p)		
Description	Cuts the child from the parent and places it the top level list of the heap structure and updates childcut values.	
Parameters	ptr p	child which is to be cut from parent. parent whose child is cut.
Return value	void	

Function Name: private void cascading cut(node p)		
Description	Performs cascading cut on the remaining structure till a false childcut value is found or root level is reached.	
Parameters	p	node from which we have to perform cascading cut.
Return value	void	

Function Name: public node removemax		
Description	Removes the node with maximum frequency from the heap and returns it.	
Parameters	none	none
Variable inside the function	maxnode children ptr	node which has the maximum value in the heap. the number of children maxnode has. points to the children of maxnode and places them in root list.
Return value	node with maximum value	

Function Name: private void consolidate		
Description	Does a pairwise combine of the remaining nodes in the heap structure.	
Parameters	none	none
Variables inside the function	degreetable roots found	hashmap that stores the nodes and degrees of the nodes seen till now. number of roots in the structure that have to combined. node that is of same degree which is found from degreetable.
Return value	void	

Conclusion

The project goal to implement Fibonacci heaps to find the most popular words has been successfully implemented for the given input files.