Fibonacci Heap

Jay R Patel

4145 1618

jaypatel1@ufl.edu

University of Florida

Keywordcounter.java

Function(s):

public static void main(String[] args)

Description:

Takes the input arguments and parses it line by line.

Calls the respective function of the heap as required by the input.

- Calls insert for first time occurring <keyword, frequency> pair
- Calls increase_key for recurring <keyword, frequency> pair
- Calls remove_max n number of times on enquiring a query for n top keyword searches
- Reinserts the removed nodes after the completion of the query for future queries
- Writes output to the file
- Stops the execution on encountering “stop”

Parameters:

args[] : Array of input arguments, (here) filename

Return value:

void

Fibonacci.java

Function(s):

public Node insert(int frequency_input, String keyword)

Description:

- Inserts a Node in the Heap with desired frequency and returns the reference of the inserted node.

Parameters:

frequency_input: Integer containing frequency of the keyword to be inserted

keyword: keyword of the Node

Return value:

Inserted node

public void meld(Node new_insert, int size_increment)

Description:

- Melds the provided list with the max pointer of the heap

Parameters:

new_insert: New node to be meld

size_increment: Increment in the size of the node

Return value:

void

public void cut(Node cut_node)

Description:

- Detaches a node from its parent and siblings

Parameters:

:

cut_node: The node to be detached

Return value:

Void

public void increase_key(Node increase_key_node, int increment)

Description:

Increases the key value of the given node by the amount of provided value.

Detaches the node if the increased value is greater than it's parent's and melds the node into root list.

Parameters:

increase_key_node: Increases the key of this node

increment: Value by which the size has to be incremented

Return value:

void

public void remove(Node remove_node)

Description:

Removes the provided node from the heap

Parameters:

remove_node: Node to be removed

Return value:

Void

public void pairwise_combine(Node root)

Description:

Used by the remove_max function to perform pairwise combine of nodes with equal degree.

Parameters:

root: Any node from the top-level list to start the pairwise combine operation

Return value:

void

public Node remove_max()

Description:

Removes and returns the max node from the heap and calls pairwise combine.

Parameters:

None

Return value:

Returns the removed node from the heap

public boolean not_empty()

Description:

Checks whether the heap is empty or not

Parameters:

None

Return value:

True if not empty. False otherwise