Advances Data Structures (COP 5536)

Fall 2016

Programming Project Report

Name: Jianing Cui

UFID: 32794301

cui2016@ufl.edu

Project Description

I implemented a system to find the n most popular hashtags appeared on social media such as Facebook or Twitter. The idea is to use a max priority structure to find out the most popular hashtags.

The following data structures are used:

1. **Max Fibonacci heap: use to keep track of the frequencies of hashtags.**
2. **Hash table: Key for the hash table is hashtag and value is pointer to the corresponding node in the Fibonacci heap**

Language/compiler: C++/g++
Developing Environment: Mac OS

Class & Function Prototypes

fiboheap.cpp

Class fibonode

Description

The element type of every node in the fiboheap

fibonode public types

1. T key;
2. int degree;
3. bool childcut;
4. fibonode<T> *left
5. fibonode<T> *right
6. fibonode<T> *child
7. fibonode<T> *parent

Description

This class is the implementation of the Fibonacci heap.

fiboheap private types

1. nt keyNum;

   description: number of nodes in the heap
2. int maxDegree;

   description: the max degree of heap
3. fibonode<T> *max;

   description: max node in the heap
4. fibonode<T> **cons;

   description: table for pairwise combine, index is the degree, element is the pointer to the subtrees

fiboheap public construct/copy/destruct

1. fiboheap();
2. ~fiboheap();

fiboheap public member functions

1. void insert(fibonode<T> *node);

   description: insert a node to the Fibonacci heap, insert as the left sibling of the max node
2. void removeMax();

   description: delete the max node from the heap, link the subtree of the max node to the root linked-list, pairwise combine the same degree subtree
3. void addNode(fibonode<T> *node1, fibonode<T> *node2);

   description: link the node1 to the left of the node2.
4. void removeNode(fibonode<T> *node);

   description: remove the node from the link list
5. fibonode<T>* getMax();

   description: get the max node
6. fibonode<T>* extractMax();

   description: it is the intermediate step of removeMax, extract the max node
7. void link(fibonode<T>* node, fibonode<T>* root);

   description: combine two tree

8. void makeCons();

   description: make memory for degree table

9. void consolidate();

   description: combine the same degree tree in the root link list

10. void renewDegree(fibonode<T> *parent, int degree);

    description: change the node's degree

11. void cut(fibonode<T> *node, fibonode<T> *parent);

    description: cut node off the parent and move it to the root link list

12. void cascadingCut(fibonode<T> *node) ;

    description: if the increase key broke the rule of max heap, then cut it, add to root linked-list. Then do the cascadingCut from its parent to the root

13. void increaseKey(fibonode<T>* node, T key);

fileio.cpp

Description

This program is responsible for file input and output, as well as call Fibonacci heap specific details

static variables

1. unordered_map<string, fibonode<int>*> hashtable;

   description: Key for the hash table is hashtag and value is pointer to the corresponding node in the Fibonacci heap

2. unordered_map<fibonode<int>*, string> reversed;

   description: store the hashtable’s value as key and key as value in a unordered_map named reversed.

3. fiboheap<int>* hb=new fiboheap<int>();

functions

1. void reversehash(std::unordered_map<std::string, fibonode<int>*> &hashtable);

   description: for every pair(key, value) in hashtable, swap the key and value and set them to a new hashmap(reversed[value]=key)

2. void ioprocess(char* filename);

   description: Processing the input file passed by main function, each line will be converted to the corresponding operation, the data structure to modify accordingly, if there is query operation, the results will be output to the file

main.cpp

Description

This program is called directly by user and the parameter is the name of input file. In main(), this parameter is passed to ioprocess().

Program structure