2022_DS_Fall_Homework 2

Notice

The deadline is 2022/12/22 23:59. Homework should be submitted as a c source file, not an executable file. In your homework, read input from `stdin` and write your output to `stdout`. The file name would like `F12345678_hw1_p1.c`.

Execution environment and Constraint.

- CPU core: 1
- Memory: 2 GB
- Execution time limit: 1 second
- C Compiler: GCC
  - compiled with `-O3 -std=c11 -Wall`
- C Standard: C11
- Use header file only from C Standard Library
- OS: Linux 22.04.1 LTS

Problem 1 : Hashing (3%)

Bloom filter consists of m bit of memory and h uniform and independent hash functions f_1, ... , f_h. Each f_i hashes a key k to an integer in the range $[1, m]$. Initially all m filter bits are zero, and the differential index and file are empty. When key k is added to the differential index, bits $f_1(k)$, ..., $f_h(k)$ of the filter are set to 1. When a query of the type "Is key k in the differential index?" is made, bits $f_1(k), ..., f_h(k)$ are examined.

Assume that initially there are n records and that u updates are make. Assume that none of these is an insert or a delete.

The probability of a filter error is

$$P(u) = (1 - 1/n)^u(1 - (1 - 1/m)^{uh})^h$$

Problem

By differentiating $P(u)$ with respect to h, show that $P(u)$ is minimized when $h = (log_e 2)m/u$. (Write a program to validate the result shown in the problem by investigating P(u) with various h's.)

Report

In the report, you need to design an experiment to show that the minimum of $P(u)$ is exist.

Problem 2 : Priority Queue (5%)

Write a C function to create an empty F-heap and support the following instructions. Each line in the input file represents one instruction.

1. `insert x val` : insert an element with key `x`
2. `extract` : **print out the minimum** in the heap and delete it
3. `delete x val` : delete the node which has key `x` and value `val`
4. `decrease x val y` : decrease the key by `y` on the node which has key `x` and value `y`
5. `quit` : terminate the program

Note that all operations must leave behind properly structured F-heaps. Your functions for (4) and (5) must perform cascading cuts.

Constraints

- $-2147483648 \leq x, \text{val} \leq 2147483647$
- $1 \leq y \leq 2147483647$
- $2 \leq n \leq 10^5$, n is number of instructions
- The key after decreasing will not exceed the bound of 32-bit signed integer.

Problem 3 : Efficient Binary Search Tree (5%)

Program the search, insert, and delete operations for AVL trees and red-black trees.

1. `search x` : Print out the ***balance factor*** (or ***color***) of the tree node if the element `x` exists. If the element `x` does not exist, print out `Not found\n`.
2. `insert x` : Add an integer `x` to the red-black/avl tree. If `x` already exists, do nothing.
3. `delete x` : Delete the element `x` if the element `x` exists.

(Note: The text book did not describe how to implement `delete` on AVL or red-black tree. Students can write your own `delete` **to satisfy the constraint** of the AVL or red-black tree.)

You are given some input files. The first line is `AVL` or `red_black`. `AVL` means implement the "instructions" by AVL-tree. `red_black` means implement the "instructions" by red-black tree.

The instuctions are insert, search ,delete and quit. Instruction quit means you should terminate your program.