Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних програмних систем

Алгоритми та складність

Завдання №2

“ Дерево порядкової статистики”

Виконав студент 2-го курсу

Групи К-29

Жутовський Глiб Євгенович

2021

**Предметна область**

Предметна область: Навчальний відділ

Об’єкти:  Групи, Студенти

Примітка: Існує множина навчальних груп. Кожна група включає в себе множину студентів.

**Завдання**

Реалізувати дерево порядкової статистики на основі червоно-чорного дерева. У вузлах цього дерева буде зберігатися інформація про групи.

**Теорія**

Червоно-чорне дерево - бінарне дерево пошуку з додатковим бітом кольору в кожному вузлу: червоний чи чорний. Має наступні властивості:

* Дерева наближено збалансовані: жоден шлях в червоно-чорному дереві не відрізняється від іншого більше ніж удвічі.
* Кожен вузол містить поля key, color, p, left, right.
* Всі вузли, що містять ключ, – внутрішні.
* Розглядатимемо значення NIL як вказівники на листя (зовнішні вузли) бінарного дерева пошуку.
* Бінарне дерево пошуку буде червоно-чорним, якщо задовольнятиме червоно-чорні властивості.

Червоно-чорні властивості:

1. Кожен вузол є або червоним, або чорним.

2. Корінь дерева – чорний.

3. Кожен лист дерева (NIL) – чорний.

4. Якщо вузол червоний, то обидва його сини чорні.

5. Для кожного вузла всі шляхи від нього до листів-потомків містять однакову кількість чорних вузлів.

Чорна висота вузла x (bh(x), black-height) – кількість чорних вузлів на шляху від вузла x (не рахуючи його самого) до листів; вона визначається однозначно.

Чорна висота дерева – чорна висота його кореня.

Дерево порядкової статистики T (order-statistic tree) – червоно-чорне дерево з додатковим інформаційним полем size[x] (розмір під дерева з коренем x).

Поклавши size[nil[T]]=0, отримуємо тотожність:

Left Rotate

y

x

x

y

Right Rotate

ɣ

α

β

β

ɣ

**Алгоритми**

**• Search (пошук елемента**) – звичайний пошук.

**• Insert (додавання елемента) –** додаємо елемент як у звичайне дерево. Викликаємо функцію InsertFixup, яка за допомогою поворотів відновлює червоно-чорні властивості.

**• Delete (видалення елемента).** Знаходимо елемент в дереві, видаляємо його, його синів підвішуємо до початкового дерева. Запускаємо DeleteFixup, яка, аналогічно до InsertFixup, за допомогою поворотів відновлює червоно-чорні властивості.

**Складність**

Всі операції потребують часу в середньому О. Висота червоно-чорного дерева з n внутрішніми вузлами не перевищує .

**Мова програмування**

С++

**Модулі програми**

#include "classes.h"

using namespace std;

//RedBlack tree class

template <class T>

class RedBlackTree

{

//Structure of node

struct RedBlackTreeNode

{

T value\_;

bool colour\_;

RedBlackTreeNode\* parent\_;

RedBlackTreeNode\* left\_;

RedBlackTreeNode\* right\_;

int size\_;

//Constructors

RedBlackTreeNode(const T& el);

RedBlackTreeNode(const RedBlackTreeNode\* node);

//Output in console

void Print(ostream& os);

};

//Overloaded operator =

RedBlackTreeNode\* operator=(const RedBlackTreeNode\* node);

RedBlackTreeNode\* root\_;

//Left rotate function

void LeftRotate(RedBlackTreeNode\* x);

//Right rotate funtion

void RightRotate(RedBlackTreeNode\* x);

//Function that repair red black tree properties after insertion

void InsertFixup(RedBlackTreeNode\* node);

//Function that repair red black tree properties after extraction

void DelFixup(RedBlackTreeNode\* p);

//Finding max element

RedBlackTreeNode\* Successor(RedBlackTreeNode\* p);

public:

//Constructor

RedBlackTree() : root\_(nullptr) {};

//Insert new element

void Insert(const T& el);

//Removing element

void Del(const T& key);

//Finding element

void Search(const T& value);

//Interactive output

void Print();

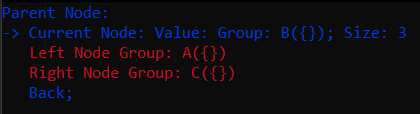
};

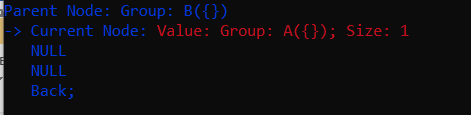
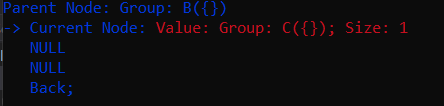
**Інтерфейс користувача**

Вхідні дані вводяться з консолі користувачем і виводяться в консоль.

**Приклад виводу програми**

Вважаємо перше дерево яке зображене в консолі є вхідними даними користувача(сині написи в дереві – чорні вузли, червоні написи – червоні вузли).

****

****  

**Тестовий приклад**

**операція insert ({D, {}})** :

Початкове дерево Після вставки

B

C

A

B

C

A

D

Перефабовка Після відновлення властивостей

B

С

А

B

С

А

D

D

**Висновки**

Дерево порядкової статистики є досить зручним у використанні через те, що воно побудовано на основі червоно-чорного дерева, а також є додаткове поле розміру під дерева, яке можна використовувати для ідентифікації конкретного елемента.

**Література**

* Лекція № 2
* Лекція № 3
* https://uk.wikipedia.org/wiki/%D0%A7%D0%B5%D1%80%D0%B2%D0%BE%D0%BD%D0%BE-%D1%87%D0%BE%D1%80%D0%BD%D0%B5\_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE