Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Завдання №3

“Розширюване дерево”

Виконав студент 2-го курсу

Групи К-29

Циганов Аркадій Олексійович

2021

**Предметна область**

Предметная область  Кафедра университета

Объекты  Преподаватели, Дисциплины

Примечание  На  кафедре  имеется    множество преподавателей.

Для  каждого преподавателя  задано  множество дисциплин.

**Завдання**

Реалізувати розширюване дерево (Splay Tree)

**Теорія**

**Розширюване дерево** (англ. *splay tree*) є двійковим деревом пошуку, у якому підтримується збалансованість. Це дерево належить до класу «саморегульованих дерев», які підтримують необхідний баланс галуження дерева, щоб забезпечити виконання операції пошуку, додавання і видалення за логарифмічний час від числа елементів, що зберігаються. Це реалізується без використання яких-небудь додаткових полів у вузлах дерева (як, наприклад, в Червоно-чорних деревах або АВЛ-деревах, де у вершинах зберігається, відповідно, колір вершини і глибина піддерева). Замість цього «розширювальні операції» (splay operation), частиною яких є повороти дерева, які виконуються при кожному зверненні до дерева. Облікова вартість з розрахунку на одну операцію з деревом складає O(log n){\displaystyle O(\log n)}ЩOO.

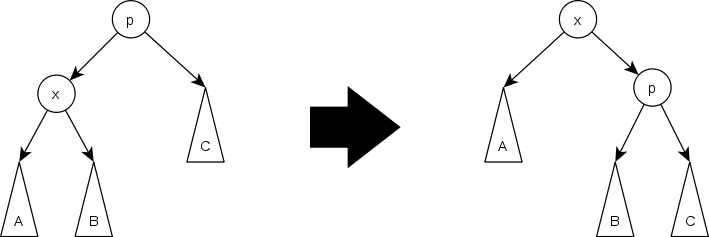
Розширюване дерево придумали Роберт Тарьян і Данієль Слейтор в 1985 році.

**Алгоритм**

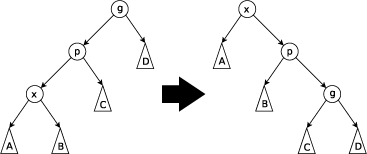
**Splay (розширення)**

Основна операція дерева. Полягає в переміщенні вершини в корінь за допомогою послідовного виконання трьох, наведених нижче, операцій: Zig, Zig-Zig і Zig-Zag. Позначимо вершину, яку хочемо перемістити в корінь за *x*, її родича — *p*, а родича *p* (якщо існує) — *g*.

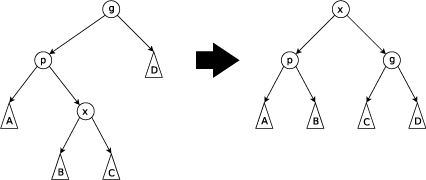
**Zig:** виконується, коли *p* є коренем. Дерево повертається по ребру між *x* та *p*. Існує лише для розбору крайнього випадку і виконується лише один раз в кінці, коли початкова глибина *x* була непарна.



**Zig-Zig:** виконується, коли *x* і *p* є лівими (або правими) синами. Дерево повертається по ребру між *p* та *x*.



**Zig-Zag:** виконується, коли *x* є правим сином, а *p* — лівим (чи навпаки). Дерево повертається по ребру між *p* та *x*, а потім — по ребру між *x* та *g*.



**Search (пошук елемента)**

Пошук виконується як в звичайному двійковому дереві пошуку. При знаходженні елементу запускаємо Splay для нього.

**Insert (додавання елемента)**

Запускаємо Splay від елементу, що додається, і підвішуємо дерева, що вийшли, за нього.

**Delete (видалення елемента)**

Знаходимо елемент в дереві, робимо Splay для нього, робимо поточним деревом Merge його дітей.

**Merge (об'єднання двох дерев)**

Для злиття дерев T1 і T2, в яких всі ключі T1 менше ключів в T2, робимо Splay для максимального елементу T1, тоді біля кореня T1 не буде правого дочірнього елемента. Після цього робимо T2 правим дочірнім елементом T1.

**Split (розділення дерева на дві частини)**

Для розділення дерева знайдемо найменший елемент, більший або рівний x і зробимо для нього Splay. Після цього відрізуємо біля кореня лівого дочірнього елемента і повертаємо 2 дерева, що вийшли.

Примітка: Розширюване дерево, надає самозмінну структуру — структуру, що характеризується тенденцією зберігати вузли, до яких часто відбувається звернення, поблизу верхівки дерева, тоді як вузли до яких звернення відбувається рідко переміщуються ближче до листя. Таким чином, час звернення до часто відвідуваних вузлів буде менший, а час звернення до рідко відвідуваних вузлів — більше середнього.

Розширюване дерево не володіє жодними явними функціями балансування, але процес скосу вузлів до кореня сприяє підтримці дерева в збалансованому вигляді.

**Складність**

Амортизована вартість будь-якої операції на розширюваному дереві становить O(log n), {\displaystyle O(\log(n)),}O(де n {\displaystyle n}n це кількість вузлів у дереві. Будь-яка операція на дереві може зайняти O(n) {\displaystyle O(n)}часу, але, як правило, також робить дерево більш збалансованим, так що з плином часу середня вартість операції є O(log n).{\displaystyle O(\log(n)).}O

**Мова програмування**

С++

**Модулі програми**

* template<typename T> class SplayTree;

Головний класс дерева, в якому реалізовані функції вставки і видалення, які викликають відповідні операції для зміни конфігурації дерева, а також функція виводу зв’язків у дереві для користувача.

**Інтерфейс користувача**

Вхідні дані задані на етапі запуску програми. На екран виводиться дерево після вставки всіх вузлів.

**Тестові приклади**

|  |  |
| --- | --- |
| **input** | **output** |
| Порядок вставки предметів із заданою кількістю годин:  2, 12, 5, 42, 1, 37 | Вигляд зв’язків в дереві (батько→син):  37 → 1  1 → 12  12 → 2  2 → 5  37 → 42 |

**Висновки**

Розширюване дерево є ефективною реалізацією бінарного дерева пошуку. Завдяки тому, що воно є збалансованим, підтримується необхідна розгалуженість дерева, тому операції пошуку в такому дереві займає O(log n) в найгіршому випадку.

**Література**

* Томас Х. Кормен и др. Алгоритмы: построение и анализ. — М. : Видавничий дім «Вільямс», 2007. — 1296 с. — ISBN 5-8459-0857-4.
* Daniel Sleator, Robert Tarjan. A data structure for dynamic trees. — Journal of Computer and System Sciences, 1983. — С. 262-391.