Київський національний університет імені Тараса Шевченка
Факультет комп`ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем
Алгоритми та складність

Завдання №2.2
“ B+ - дерево”
Виконав студент 2-го курсу
Групи ІПС-21
Ляшенко Матвій Олексійович

2025

Завдання

Реалізувати B+ - дерево

Теорія

B-дерево — це узагальнення бінарного дерева пошуку, яке дозволяє ефективно зберігати великі обсяги впорядкованих даних. Його головна відмінність — **висока ступінь розгалуження**, коли один вузол може мати десятки або навіть тисячі нащадків.

Кожне B-дерево T із коренем root[T] має такі властивості:

- Кожен вузол xx містить:
  - n[x] — кількість ключів у вузлі;
  - ключі key1[x]≤key2[x]≤⋯≤keyn[x][x];
  - логічне поле leaf[x], яке визначає, чи є вузол листком.
- Кожен внутрішній вузол xx має n[x]+1 вказівників на дочірні вузли: c1[x],c2[x],…,cn[x]+1[x].
- Ключі у вузлі розділяють піддіапазони ключів у дочірніх піддеревах. Якщо ki — ключ у піддереві з коренем ci[x], то: k1≤key1[x]≤k2≤key2[x]≤⋯≤keyn[x][x]≤kn[x]+1
- Усі листки знаходяться на одній і тій самій глибині h, тобто дерево є **ідеально збалансованим**.
- Мінімальна та максимальна кількість ключів у вузлі визначається цілим параметром t≥2 — **мінімальним степенем**:
  - Вузол (крім кореня) містить принаймні t−1 ключів (тобто має щонайменше t нащадків);
  - Кожен вузол містить не більше ніж 2t−1 ключів (максимум 2t нащадків);
  - Вузол вважається **повним**, якщо має рівно 2t−1 ключів.

B+-дерево є варіацією B-дерева, яка має додаткові властивості:

- **Усі істинні значення ключів** містяться лише у листках.
- Внутрішні вузли зберігають **тільки ключі-роздільники**, які використовуються для маршрутизації пошуку.
- **Усі листки зв'язані в двосторонній список**, що забезпечує ефективну побудову **послідовного обходу**.
- Пошук, вставка і видалення ключів **завжди закінчуються в листках**, що спрощує реалізацію.
- Така структура забезпечує **ефективний доступ до діапазонів** значень і полегшує операції сортування.
- Недоліком є **підвищене використання пам'яті**, порівняно зі звичайним B-деревом, через додаткові зв'язки між листками.

Предметна область

Варіант №15; Тип даних T5

Тип даних: Комплексні числа з цілими компонентами.

Порівняння: спочатку за модулем, потім — за дійсною частиною.

Алгоритм

Видалення
Видалення ключа D з B+ дерева відбувається у два основні етапи:

1. **Видалення з листка** — ключ видаляється зі списку значень, розташованого в листовому вузлі.
2. **Оновлення внутрішнього вузла** — якщо ключ D також присутній як **роздільник** у внутрішньому вузлі, його необхідно замінити на новий ключ K, який продовжує впорядкування.

Як обирається новий ключ K?

- Ключ KK має задовольняти умову:
  M<K<I
  де:
  - M — попередній ключ у внутрішньому вузлі (зліва від D);
  - I — наступний ключ (праворуч від D).
- Кандидат на роль K обирається зі **того самого листа**, з якого було видалено D. Якщо там недостатньо ключів:
  - шукаємо праворуч у сусідньому листі (якщо ключі занадто великі);
  - або ліворуч (якщо занадто малі).

Вставка

1. Спочатку шукаємо **позицію у листі**, проходячи ключі зліва направо.
2. Вставляємо новий ключ K **після першого більшого ключа** в листі.
3. Перевіряємо, чи виконуються **властивості B+ дерева**:
   - чи належить новий ключ інтервалу в батьківському вузлі?
4. Якщо інтервал порушено:
   - розділяємо листовий вузол;
   - вносимо новий роздільник у батьківський вузол;
   - можливо, рекурсивно повторюємо розділення вгору.

Складність

Усі операції з деревом займають $O(h)$ де h - висота дерева для якої справедлива нерівність h <=logt((n+1)/2), де t степінь дерева (мінімальна кількість піддерев) то складність не перевищує $O(log\ t\ n)$

Мова програмування

C++

Модулі програми

1. Модуль ComplexInt
    - Описує користувацький тип комплексних чисел з цілими компонентами (int real, int imag).
    - Реалізує порівняння комплексних чисел за модулем (довжиною вектора) та за дійсною частиною при рівних модулях.
    - Містить перевизначення операторів <, ==, >, <=, >=, !=, а також << для виводу в консоль.
    - Обчислення модуля виконується один раз у конструкторі та кешується для підвищення продуктивності.
2. Модуль BPlusTree - Реалізує шаблонну структуру B+ дерева, яка підтримує будь-який тип, що має оператор <.
    - Складається з:
        - BPlusNode<T> — вузол дерева (листковий або внутрішній);
        - BPlusTree<T> — основний клас з методами вставки, видалення, виводу.
    - Ключові методи:
        - insert() — вставка нового елемента;
        - remove() — видалення з листка;
        - print() — вивід дерева по рівнях;
        - printSorted() — вивід усіх елементів у впорядкованому вигляді.
3. Модуль example_ComplexInt() - Забезпечує інтерфейс користувача через консоль. Містить **меню** з такими пунктами:
    - вставка нового комплексного числа;
    - видалення;
    - перегляд структури дерева;
    - вивід відсортованого списку
    - працює в циклі до вибору опції "Exit"

Інтерфейс користувача

Вхідні дані вводяться з консолі користувачем і виводяться в консоль.

Тестовий приклад

====== MENU ======
1. Insert complex number
...
Enter real and imaginary part: 3
4
Enter real and imaginary part: 1
1
Enter real and imaginary part: 0
0
Enter real and imaginary part: 2
2
Enter real and imaginary part: -3
4

...

Sorted list:
1) 0+0i
2) 1+1i
3) 2+2i
4) -3+4i
5) 3+4i

Remove complex number: 3
4

After removal:
1) 0+0i
2) 1+1i
3) 2+2i
4) -3+4i

Висновки

У ході виконання лабораторної роботи було реалізовано структуру B+ дерева, адаптовану під індивідуальний варіант з використанням комплексних чисел з цілочисельними компонентами як ключів. У цілому, структура B+ дерева добре підходить для задач, де необхідна швидка вставка, видалення та впорядкований доступ до даних, і є ефективною у контексті роботи з великими обсягами інформації.

Література

- https://habr.com/ru/company/sberbank/blog/413749/
- Лекція № 4