Cập nhật tháng 8 năm 2024

[Bài đọc] Thuật toán tìm kiếm nhị phân

Tìm kiếm là một kỹ thuật cơ bản trong khoa học máy tính để xác định vị trí của một phần tử trong một danh sách. Hai thuật toán phổ biến nhất để tìm kiếm là **tìm kiếm tuần tự** và **tìm kiếm nhị phân.**

1. **Tìm kiếm nhị phân (Binary Search)**

- Khái niệm
    - Tìm kiếm nhị phân là một thuật toán tìm kiếm hiệu quả áp dụng trên danh sách **đã được sắp xếp**
    - Thuật toán hoạt động bằng cách chia đôi danh sách và so sánh phần tử ở giữa với giá trị cần tìm
    - Tùy thuộc vào kết quả so sánh, phần danh sách có thể bị loại bỏ một nửa, và quá trình này lặp lại cho đến khi tìm thấy phần tử hoặc danh sách rỗng
- Đặc điểm
    - Độ phức tạp thời gian (Time Complexity):
        - Trường hợp tốt nhất: O(1) (khi phần tử nằm ở giữa)
        - Trường hợp xấu nhất: O(log n) (do danh sách bị chia đôi sau mỗi lần kiểm tra)
    - Yêu cầu danh sách **đã được sắp xếp** trước khi áp dụng
- Ưu điểm
    - Nhanh hơn nhiều so với tìm kiếm tuần tự cho danh sách lớn
    - Hiệu quả khi xử lý dữ liệu đã được sắp xếp
- Nhược điểm
    - Không áp dụng được cho danh sách không sắp xếp
    - Yêu cầu thao tác sắp xếp trước khi tìm kiếm, có thể tốn chi phí
- Ví dụ minh họa

```java
public static int binarySearch(int[] arr, int key)  1 usage  new *
{
    int left = 0, right = arr.length - 1;
    while (left <= right)
    {
        int mid = left + (right - left) / 2;
        if (arr[mid] == key)
        {
            return mid; // Phần tử được tìm thấy
        }
        if (arr[mid] < key)
        {
            left = mid + 1; // Tìm trong nửa phải
        }
        else
        {
            right = mid - 1; // Tìm trong nửa trái
        }
    }
    return -1; // Không tìm thấy
}

public static void main(String[] args)  new *
{
    int[] arr = {2, 3, 5, 6, 8}; // Mảng đã sắp xếp
    int key = 6;
    int result = binarySearch(arr, key);
    System.out.println("Phần tử tìm thấy ở chỉ số: " + result);
}
```

2. **So sánh giữa Tìm kiếm tuần tự và Tìm kiếm nhị phân**

Tiêu chí	**Tìm kiếm tuần tự**	**Tìm kiếm nhị phân**
Danh sách sắp xép	Không yêu cầu	Yêu cầu đã sắp xếp
Hiệu xuất	Chậm với danh sách lớn	Nhanh hơn, đặc biệt với danh sách lớn
Độ phức tạp tốt nhất	O(1)	O(1)
Độ phức tạp xấu nhất	O(n)	O(log n)
Triển khai	Đơn giản	Phức tạp hơn
Ứng dụng	Dữ liệu nhỏ, không sắp xếp	Dữ liệu lớn, đã sắp xếp

3. **Khi nào nên sử dụng**

- Tìm kiếm tuần tự:
    - Khi danh sách nhỏ và chưa được sắp xếp
    - Khi cần giải pháp đơn giản và nhanh chóng để kiểm tra dữ liệu

- Tìm kiếm nhị phân:
    - Khi danh sách đã được sắp xếp và bạn cần hiệu suất cao
    - Khi làm việc với dữ liệu lớn và tìm kiếm thường xuyên

Tài nguyên đọc thêm: https://www.geeksforgeeks.org/binary-search-in-java

Danh sách các bài học