I. Bài tập tìm kiếm

*1. Trình bày ngắn gọn tư tưởng các giải thuật tìm kiếm, các giải thuật này có thể được vận dụng trong các trường hợp nào, cho ví dụ minh họa*

**Tìm kiếm tuyến tính**

*Tư tưởng:* Duyệt qua tất cả các phần tử của dãy, trong quá trình duyệt nếu tìm thấy phần tử có giá trị khoá bằng với giá trị khoá cần tìm kiếm thì trả về trị trí của phần tử tìm được. Ngược lại, nếu duyệt tới hết dãy mà vẫn không có phần tử thoả mãn yêu cầu trả về giá trị mặc định đó.

Giải thuật này được vận dụng trong trường hợp ít bộ testcase, giới hạn nhỏ.

Ví dụ minh hoạ: Dãy gồm phần tử ( ≤ ), hoặc giới hạn đoạn

**Tìm kiếm nhị phân**

*Tư tưởng:* Giải thuật tìm kiếm nhị phân là giải thuật dùng để tìm kiếm phần tử trong một dãy đã được sắp xếp. Trong mỗi bước, ta tiến hành so sánh phần tử cần tìm với phần tử nằm ở chính giữa dãy. Nếu hai phần tử bằng nhau thì thao tác tìm kiếm thành công và giải thuật kết thúc. Nếu chúng không bằng nhau thì tuỳ vào phần tử nào lớn hơn, giải thuật lặp lại bước so sánh trên với nửa đầu hoặc nửa sau của dãy và cứ tiếp tục như thế.

Giải thuật này được vận dung trong trường hợp nhiều bộ testcase, giới hạn lớn.

Ví dụ minh hoạ: bộ tests, mỗi bộ test gồm phần tử (, hoặc giới hạn

*2. Hãy trình bày các ưu, nhược điểm của các giải thuật tìm kiếm?*

**Tìm kiếm tuyến tính**

*Ưu điểm:* Dễ cài đặt, ngắn gọn, dễ hiểu, chạy nhanh với bộ test nhỏ, giới hạn nhỏ.

*Nhược điểm:* Không thể chạy được nhiều bộ testcase, giới hạn lớn.

**Tìm kiếm nhị phân**

*Ưu điểm:* Dễ cài đặt, ngắn gọn, dễ hiểu, chạy nhanh với bộ test lớn, giới hạn lớn

*Nhược điểm:* Điều kiện tiên quyết của giải thuật này là tập dữ liệu đã được sắp xếp trước đó, vì vậy nó không phù hợp khi tập dữ liệu thường xuyên thay đổi.

*3. Giả sử cho một dãy số nguyên N phần tử, các phần tử có thứ tự tăng dần và được yêu cầu áp dụng giải thuật tìm kiếm tuyến tính để tìm kiếm phần tử trong dãy, hãy thực hiện các yêu cầu sau:*

*a. Cài đặt theo yêu cầu trên và đánh giá về số lần so sánh trong các trường hợp tìm kiếm phần tử khi cho phần tử cần tìm kiếm nằm ở đầu dãy, cuối dãy và giữa dãy*

**Cài đặt:**

bool linearSearch(vector <int> array, int n, int X){

for (int i = 0; i < n; ++i){

if (array[i] == X){

return true;

}

}

return false;

}

Khi phần tử cần tìm kiếm nằm ở đầu dãy, độ phức tạp sẽ là

Khi phần tử cần tìm kiếm nằm ở giữa dãy, độ phức tạp sẽ là

Khi phần tử cần tìm kiếm nằm ở cuối dãy, độ phức tạp sẽ là

*b. Cải tiến giải thuật tìm tuyến tính để phù hợp với giả thiết đã cho.*

**Cải tiến giải thuật: Sử dụng giải thuật tìm kiếm nhị phân**

bool binarySearch(vector <int> array, int n, int X){

sort(array.begin(), array.end());

int L = 0, R = n - 1;

while (L <= R){

int mid = (L + R) >> 1;

if (array[mid] == X){

return true;

}

if (array[mid] < X){

L = mid + 1;

} else {

R = mid - 1;

}

}

return false;

}

*c. Đánh giá và so sánh tính hiệu quả về bộ nhớ và số giao tác khi thực hiện hai giải thuật trong câu a và câu b.*

|  |  |  |
| --- | --- | --- |
|  | Tìm kiếm tuyến tính | Tìm kiếm nhị phân |
| Bộ nhớ | Bằng nhau | Bằng nhau |
| Số thao tác | Bằng chỉ số vị trí của phần tử cần tìm (ví dụ ở vị trí thứ i thì so sánh i lần) | So sánh lần, nhanh hơn rất nhiều |

*4. Giả sử cho một dãy số nguyên N phần tử, các phần tử có thứ tự giảm dần và được yêu cầu áp dụng giải thuật tìm kiếm nhị phân để tìm kiếm phần tử trong dãy, hãy thực hiện các yêu cầu sau:*

*a. Cài đặt theo yêu cầu trên bằng cách dùng kỹ thuật đệ qui và không dùng kỹ thuật đệ qui.*

**Kỹ thuật đệ qui:**

bool binarySearch(vector <int> array, int L, int R, int X){

if (R >= L){

int mid = (L + R) >> 1;

if (array[mid] == X) return 1;

if (array[mid] < X) binarySearch(array, mid + 1, R, X);

else binarySearch(array, L, mid, X);

}

return 0;

}

**Kỹ thuật không dùng đệ quy:**

bool binarySearch(vector <int> array, int n, int X){

int L = 0, R = n - 1;

while (L <= R){

int mid = (L + R) >> 1;

if (array[mid] == X){

return true;

}

if (array[mid] < X){

R = mid - 1;

} else {

L = mid + 1;

}

}

return false;

}

*b. Đánh giá về số lần so sánh trong các trường hợp tìm kiếm phần tử khi cho phần tử cần tìm kiếm nằm ở đầu dãy, cuối dãy và giữa dãy trong câu a.*

Khi phần tử cần tìm kiếm ở đầu dãy hoặc cuối dãy, độ phức tạp sẽ là

Khi phần tử cần tìm kiếm ở giữa dãy, độ phức tạp sẽ là

*5. Giả sử cho một dãy số nguyên gồm M phần tử (100<M< 30.000), sau đó chọn ngẫu nhiên trong dãy M một số nguyên K, áp dụng đồng thời 2 giải thuật tìm kiếm nhị phân và tuyến tính để tìm K trong M. Hãy cho nhận xét về thời gian thực hiện nếu phát sinh ngẫu nhiên 100 lần M và K khác nhau và thực hiện hai giải thuật nêu trên để tìm K trong M.*

Thời gian thực hiện giải thuật tìm kiếm tuyến tính: , sẽ chậm với M lớn

Thời gian thực hiện giải thuật tìm kiếm nhị phân: ), sẽ chạy rất nhanh.

*6. Giả sử cho một dãy số nguyên N phần tử (100 < N < 10.000).*

*a. Hãy cho biết thời gian thực hiện giải thuật tìm kiếm tuyến tính để tìm kiếm một phần tử K trong dãy N khi áp dụng kỹ thuật phần tử “lính canh” và khi không dùng kỹ thuật phần tử “lính canh”.*

Thời gian thực hiện giải thuật tìm kiếm tuyến tính khi áp dụng kỹ thuật phần tử "lính canh" để tìm kiếm một phần tử K trong dãy N là O(N), còn khi không dùng kỹ thuật phần tử "lính canh" thì cũng là O(N).

*b. Thời gian thực hiện giải thuật khi áp dụng kỹ thuật “lính canh” và khi không dùng kỹ thuật “lính canh” trong câu a có tỷ lệ tuyến tính với nhau khi tăng hoặc giảm số phần tử N không? Vì sao?*

Thời gian thực hiện giải thuật khi áp dụng kỹ thuật "lính canh" và khi không dùng kỹ thuật "lính canh" trong câu a có tỷ lệ tuyến tính với nhau không phụ thuộc vào số phần tử N. Vì thời gian tìm kiếm tuyến tính phụ thuộc vào số lượng phần tử cần phải duyệt qua, nên thời gian tìm kiếm sẽ tuyến tính với số phần tử của dãy N. Do đó, dù có tăng hoặc giảm số phần tử N thì thời gian tìm kiếm tuyến tính sẽ không thay đổi tỷ lệ với số phần tử của dãy.

*7. Giả sử cho một mảng 2 chiều các số nguyên có kích thước M x N (100 < M < 10.000), (100 < N < 20.000). Hãy cho biết thời gian thực hiện giải thuật tìm kiếm tuyến tính để tìm kiếm một phần tử K trong mảng 2 chiều trên, khi áp dụng kỹ thuật phần tử “lính canh” và khi không dùng kỹ thuật phần tử “lính canh”.*

Thời gian thực hiện giải thuật tìm kiếm tuyến tính khi áp dụng kỹ thuật phần tử "lính canh" để tìm kiếm một phần tử K trong dãy N là O(M \* N), còn khi không dùng kỹ thuật phần tử "lính canh" thì cũng là O(M \* N).

*8. Giả sử sử dụng hàm Random trong C/C++ để phát sinh ngẫu nhiên dãy 10.000 số nguyên và lưu trong file X.*

*a. Hãy áp dụng giải thuật tìm tuyến tính để tìm kiếm số nguyên K trong file X (K được phát sinh ngẫu nhiên)*

**Cài đặt:**

bool linearSearch(vector <int> array, int n, int K){

for (int i = 0; i < n; ++i){

if (array[i] == K){

return true;

}

}

return false;

}

*b. Cho nhận xét về thời gian tìm kiếm trong câu a sẽ như thế nào, nếu thực hiện 5 lần chạy với dãy số nguyên có 100, 1.000, 5.000, 10.000, 50.000 phần tử trong mỗi lần chạy*

- Thời gian thực hiện sau 5 lần sẽ tăng lên vì độ phức tạp của thuật toán tìm kiếm tuyến tình là O(N), với N càng lớn thì độ phức tạp và thời gian càng lớn. Tuy nhiên trong trường hợp kết quả nằm gần với vị trí duyệt thì thời gian sẽ nhanh hơn.

*9. Giả sử sử dụng hàm Random trong C/C++ để phát sinh ngẫu nhiên dãy 10.000 số nguyên và lưu trong file X.*

*a. Hãy áp dụng giải thuật tìm nhị phân để tìm kiếm số nguyên K trong file X (K được phát sinh ngẫu nhiên).*

**Cài đặt:**

bool binarySearch(vector <int> array, int n, int K){

sort(array.begin(), array.end());

int L = 0, R = n - 1;

while (L <= R){

int mid = (L + R) >> 1;

if (array[mid] == K){

return true;

}

if (array[mid] < K){

L = mid + 1;

} else {

R = mid - 1;

}

}

return false;

}

*b. Nhận xét về thời gian tìm kiếm trong câu a sẽ như thế nào, nếu thực hiện 5 lần chạy với dãy số nguyên có 100, 1.000, 5.000, 10.000, 50.000 phần tử trong mỗi lần chạy.*

- Thời gian chạy chương trình sẽ xấp xỉ bằng nhau vì độ phức tạp thuật toán là O(log2(N)).