Phân tích bài toán Codeforces 1451B

Non-Substring Subsequence

Algorithmic Coach

2026

Lời chào đầu

Mục tiêu

Chào bạn! Tôi là **Algorithmic Coach**. Chúng ta sẽ cùng "mổ xẻ"bài **Codeforces 1451B**. Chúng ta sẽ không vội vàng viết code ngay mà sẽ bóc tách lớp vỏ bài toán để tìm ra bản chất.

Bước 1: Phẫu thuật đề bài (Deconstruction)

Yêu cầu cốt lõi:

- **Dữ liệu:** Chuỗi S chỉ gồm ký tự '0' và '1'.
- **Truy vấn:** Đoạn con từ vị trí l đến r (gọi là $S[l..r]$).
- **Mục tiêu:** Kiểm tra tồn tại một **dãy con** (subsequence) bằng $S[l..r]$ nhưng **không phải là đoạn liền mạch**.

Ẩn dụ hóa

Tưởng tượng chuỗi hạt màu. Bạn chọn một đoạn hạt liên tiếp. Bạn cần tìm cách "nhặt"các hạt khác (giữ đúng thứ tự) nhưng phải "nhảy cóc"ít nhất một lần.

Mảnh ghép 1: Bản chất của sự "Nhảy cóc"

Để dãy con không phải chuỗi con liền mạch, ta cần ít nhất một "khoảng trống".

Bẫy (Trap)

Không nhất thiết phải thay đổi các ký tự ở giữa. Chỉ cần thay đổi "điểm bắt đầu"hoặc "điểm kết thúc"là đủ phá vỡ tính liên tiếp.

Thử thách tư duy: Cho chuỗi 001010. Xét đoạn $S[2..4]$ (là 010). Nếu chọn các vị trí $\{2, 3, 6\}$, dãy thu được là 010.

Mảnh ghép 1: Bản chất của sự "Nhảy cóc"

Để dãy con không phải chuỗi con liền mạch, ta cần ít nhất một "khoảng trống".

Bẫy (Trap)

Không nhất thiết phải thay đổi các ký tự ở giữa. Chỉ cần thay đổi "điểm bắt đầu"hoặc "điểm kết thúc"là đủ phá vỡ tính liên tiếp.

Thử thách tư duy: Cho chuỗi 001010. Xét đoạn $S[2..4]$ (là 010). Nếu chọn các vị trí $\{2, 3, 6\}$, dãy thu được là 010.

- Dãy này thỏa mãn vì giữa hạt số 3 và 6 có "khoảng trống"(hạt 4, 5 bị bỏ qua).

Mảnh ghép 2: Điều kiện tối thiểu để "Nhảy cóc"

Chiến thuật đơn giản nhất:

1. Giữ nguyên hầu hết các hạt trong đoạn $S[l..r]$.
2. Thay thế **hạt đầu tiên** ($S[l]$) hoặc **hạt cuối cùng** ($S[r]$) bằng hạt cùng màu ở xa hơn.

Thử thách tư duy: Chuỗi 111000, truy vấn $S[3..4]$ (10).

- Vị trí $l = 3$ (ký tự '1'). Phía trước $(1, 2)$ có hạt '1' không?
- Vị trí $r = 4$ (ký tự '0'). Phía sau $(5, 6)$ có hạt '0' không?

Mảnh ghép 2: Điều kiện tối thiểu để "Nhảy cóc"

Chiến thuật đơn giản nhất:

1. Giữ nguyên hầu hết các hạt trong đoạn $S[l..r]$.
2. Thay thế **hạt đầu tiên** ($S[l]$) hoặc **hạt cuối cùng** ($S[r]$) bằng hạt cùng màu ở xa hơn.

Thử thách tư duy: Chuỗi 111000, truy vấn $S[3..4]$ (10).

- Vị trí $l = 3$ (ký tự '1'). Phía trước $(1, 2)$ có hạt '1' không?
- Vị trí $r = 4$ (ký tự '0'). Phía sau $(5, 6)$ có hạt '0' không?

Kết luận

Chuẩn luôn! Chỉ cần mượn được ít nhất một hạt nằm ngoài phạm vi $[l, r]$ cùng màu, ta sẽ tạo được bước nhảy.

Mảnh ghép 3: Chốt hạ thuật toán

Quy luật đơn giản cho mỗi truy vấn (l, r):

1. **Kiểm tra phía trước:** Có ký tự nào giống $S[l]$ ở vị trí từ 1 đến $l-1$ không?
2. **Kiểm tra phía sau:** Có ký tự nào giống $S[r]$ ở vị trí từ $r+1$ đến n không?

Kết quả

Nếu **một trong hai** đúng $\rightarrow$ `YES`. Ngược lại $\rightarrow$ `NO`.

Câu hỏi chốt hạ: Chuỗi `001100`, truy vấn $S[3..4]$ (`11`).

Mảnh ghép 3: Chốt hạ thuật toán

Quy luật đơn giản cho mỗi truy vấn (l, r):

1. **Kiểm tra phía trước:** Có ký tự nào giống $S[l]$ ở vị trí từ 1 đến $l-1$ không?
2. **Kiểm tra phía sau:** Có ký tự nào giống $S[r]$ ở vị trí từ $r+1$ đến n không?

Kết quả

Nếu **một trong hai** đúng $\rightarrow$ `YES`. Ngược lại $\rightarrow$ `NO`.

Câu hỏi chốt hạ: Chuỗi `001100`, truy vấn $S[3..4]$ (`11`).

- Đáp án: **NO**. Vì phía trước chỉ có `00`, phía sau chỉ có `00`. Không có hạt '1' nào để "mượn".

Tổng kết thuật toán & Mã giả

```
Cho   m i    b    test:
     c    n, q  v    c h u i  S
   V i    m i   truy   v n   (l, r):
     B i n  can_jump = false
    //  K i m  tra  ph  a   t r    c   (l-1  v    string index 0)
     D u y t    t    i = 0    n    l-2:
        N u   S[i] == S[l-1]  t h    can_jump = true

    //  K i m  tra  ph  a  sau
     D u y t    t    i = r    n    n-1:
        N u   S[i] == S[r-1]  t h    can_jump = true

     N u   can_jump  l    true: In "YES"
     N g    c    l i  : In "NO"

```

Listing 1: Mã giả logic bài toán

Tại sao không dùng Quy hoạch động (DP)?

Dùng DP có thể là "dao mổ trâu giết gà":

- DP giúp trả lời: Có thể tạo tiền tố độ dài i bằng cách dùng hạt đến vị trí j.
- Trạng thái: $dp[i][j]$ là boolean.

Nhận xét quan trọng

Nếu không thể thay đổi hạt ở đầu (l) và không thể thay đổi hạt ở cuối (r), thì việc thay đổi ở giữa không giúp tạo ra "dãy con không liên mạch".

Gợi ý tối ưu: Sử dụng `first[char]` và `last[char]` để kiểm tra trong $O(1)$ sau khi tiền xử lý $O(n)$.

Kết thúc

Bạn có muốn tôi hỗ trợ viết mã C++ hoàn chỉnh hay bạn muốn tự thử sức trước?

Chúc bạn thành công!