Phân tích và thiết kế thuật toán - CS112.Q11.KHTN - BTVN Nhóm 5

Sinh viên thực hiện (Nhóm 4):

- Bảo Quý Định Tân
- Lê Văn Thức

Gọi a_i là chuỗi chứa 2 loại giao thức của máy tính thứ i.

Subtask 1: $n, q \leq 1000$

Ý tưởng:

- Ở mỗi truy vấn, nếu ta kiểm tra thấy được ở a_i và a_j có chung một ký tự giống nhau thì đáp án là $|i - j|$ luôn vì đi thẳng từ i đến j trực tiếp là cách đi tối ưu nhất rồi (dễ thấy).
- Nếu không, ta có thể chứng minh rằng chỉ cần đi tối đa là 1 bước trung gian trước khi i có thể đến được j.
- Không mất tính tổng quát, giả sử $a_i = \text{"BG"}$, $a_j = \text{"RY"}$ thì qua những loại còn lại: $\text{"BR"}, \text{"BY"}, \text{"GR"}, \text{"GY"}$ ta đều có thể sử dụng để đi trung gian một bước giữa trước khi từ i đến j hoặc từ j đến i.
- Cho nên hướng tiếp cận trong trường hợp i không thể đến trực tiếp j là ta duyệt qua lần lượt các vị trí k $(1 \leq k \leq n)$ và kiểm tra nếu a_k khác với cả a_i và a_j thì ta có thể sử dụng vị trí k này làm bước trung gian với chi phí là $|i - k| + |j - k|$, và cứ vậy tìm tuần tự để lấy được kết quả tốt nhất.
- Còn nếu vẫn không tìm được vị trí trung gian nào thỏa mãn thì đáp án là -1.

Độ phức tạp:

- Thời gian:
  - Tham số kích thước đầu vào: n và q.
  - Thao tác cơ bản: duyệt qua tìm các bước trung gian, gồm n vị trí cần duyệt và duyệt q lần.
  - Biểu diễn thời gian chạy theo tham số đầu vào: $T(n, q) = q \times O(n)$.
  - Đơn giản hóa biểu thức: $T(n, q) = O(nq)$.
- Không gian:
  - Có mảng a chứa n chuỗi: $O(n)$.
  - Các biến tạm: $O(1)$.
  - Vậy: $S(n) = O(n)$.

Subtask 2: $n, q \le 10^5$

Ý tưởng:

- Ở mỗi truy vấn, như subtask trên, nếu từ i có thể đi trực tiếp đến j ta vẫn đi luôn, và lấy $|i-j|$ làm kết quả.
- Còn không, thay vì phải duyệt qua lại cả mảng a để tìm vị trí trung gian, ta sẽ thực hiện thông minh hơn: Vị trí trung gian k có 3 trường hợp xảy ra, giả sử $i \le j$:
  - 1. Nằm ở giữa i và j : Ta chỉ cần xem coi có vị trí k $(i \le k \le j)$ nào thỏa mãn a_k khác a_i và a_j. Nếu có đáp án này cũng là tối ưu nhất: $|i-k|+|j-k| = i-j$.
  - 2. Nằm ở bên trái i $(k \le i)$: Như trên nhưng ta nhận xét rằng để càng tối ưu thì vị trí k cần gần i nhất có thể, đáp án thu được là $|i-k|+|j-k| = i-k+j-k$.
  - 3. Nằm ở bên phải j $(j \le k)$: Như trên nhưng ta nhận xét rằng để càng tối ưu thì vị trí k cần gần j nhất có thể, đáp án thu được là $|i-k|+|j-k| = k-i+k-j$.
- Sau đó ta chỉ lấy ra kết quả tốt nhất trong 3 trường hợp trên, nhưng làm sao để tìm được vị trí k nhanh mà không cần phải duyệt qua hết cả mảng a.
- Ta nhận thấy rằng số lượng loại chuỗi khác nhau rất nhỏ, chỉ có 6 loại tổng cộng $\text{"BG"}, \text{"BR"}, \text{"BY"}, \text{"GR"}, \text{"GY"}, \text{"RY"}$, mà bỏ đi 2 loại của a_i và a_j thì chỉ còn 4 loại có thể xảy ra cho vị trí k thôi.
- Nên với mỗi loại trong 6 loại trên kia, giả sử loại "BG" ta sẽ chuẩn bị trước 2 mảng:
  - $nearestLeft[\text{"BG"}][i]$ = $\text{vị trí ở phía bên trái gần } i \text{ nhất có loại "BG"}$.
  - $nearestRight[\text{"BG"}][i]$ = $\text{vị trí ở phía bên phải gần } i \text{ nhất có loại "BG"}$.
- Với mỗi trường hợp trên, ta duyệt qua 4 loại mà khác a_i và a_j,giả sử là "BG":
  - 1. Ta xét coi $k = nearestRight[\text{"BG"}][i] \le j$ hay không?
  - 2. Ta tìm coi $k = nearestLeft[\text{"BG"}][i] \ge 1$ hay không?
  - 3. Ta tìm coi $k = nearestRight[\text{"BG"}][j] \le n$ hay không?
- Nếu thỏa mãn (vị trí k hợp lệ trong mỗi trường hợp) thì ta lấy đó để cập nhật đáp án.

Độ phức tạp:

- Thời gian:

- Tham số kích thước đầu vào: n và q.
- Thao tác cơ bản: Với mỗi loại trong 6 loại trên, ta chuẩn bị 2 mảng. Với mỗi truy vấn chỉ tính toán trong $O(1)$.
- Biểu diễn thời gian chạy theo tham số đầu vào: Với 6 loại, mỗi loại chuẩn bị 2 lần là $12n$ thao tác hay $O(n)$. Mỗi truy vấn tối đa 4 loại $\times$ 3 trường hợp, toàn bộ là tính toán và so sánh nên là $O(1)$. Tổng lại là $T(n,q) = O(n) + q \times O(1) = O(n+q)$.
- Đơn giản hóa biểu thức: $T(n,q) = O(n+q)$.

- Không gian:
  - Có mảng a chứa n chuỗi và với mỗi loại có thêm 2 mảng: $n + 2 * 6 * n$ hay $O(n)$.
  - Các biến tạm: $O(1)$.
  - Vậy: $S(n) = O(n)$.