Lời giải cho các bài thi thử ngày 2 trường THPT chuyên Lam Sơn
Lê Xuân Mạnh manhlx.300695@gmail.com

PUZZLAND

Để làm được bài này các bạn cần phải biết quy hoạch động trạng thái, là một dạng quy hoạch động khá phổ biến.

Gọi F[vertex][state] = true/false là khả năng đi từ đỉnh vertex đến đỉnh t, và đi qua các đỉnh là các bit 1 trong trạng thái state (bit thứ i bằng 1 tức là đỉnh thứ (i+1) đã được đi qua trên đường đi từ vertex đến t. Ví dụ state = 19 = 010011 tức là các đỉnh thứ 1, 2, 5 được đi qua. (lưu ý bit được đánh số từ 0).

Vỡi mỗi trạng thái F[vertex][state], ta tìm các đỉnh kề của vertex để cập nhật cho trạng thái tiếp theo. Nếu v là đỉnh kề của vertex và bit thứ v trong state bằng 0, cập nhật lại F[v][state’] = true trong đó state’ = state + (2 ^ (v-1)).

Từ trạng thái của mảng F chúng ta có thể tìm ra đường đi với thứ tự từ điển nhỏ nhất.
Bắt đầu từ trạng thái state = 2 ^ n - 1 (cả n bit đều bằng 1), và đi bắt đầu từ đỉnh s = 1 -> kiểm tra F[1][state] = true/false.
Trong khi state > 0 tìm đỉnh có giá trị nhỏ nhất (giá trị alphabet) sao cho đỉnh đó nằm trong trạng thái state, giả sử đó là đỉnh v -> kiểm tra F[v][state] = true/false, giảm giá trị của state bằng cách loại bỏ bit thứ (v-1).

Cứ làm như vây cho đến khi đi được từ s đến t ta tìm đc đường đi có thứ tự từ điển nhỏ nhất.

PALIN

Đầu tiên ta tạo mảng F[i][j] = true/false (1 <= i <= j <= n): xâu con từ i đến j có là xâu đối xứng hay không. Độ phức tạp: N^2.
Tiếp theo ta tạo mảng Next[i][t] = j trong đó: j >= i và xâu con j -> j + l[t] - 1 là một xâu đối xứng. Nếu không có xâu nào thì Next[i][t] = n + 1. Mảng Next[i][t] là vị trí gần i nhất (>=i) sao cho tồn tại 1 xâu con độ dài l[t] là xâu đối xứng.

Tiếp theo chỉ cần với mỗi vị trí i (1 <= i <= n), ta tìm vị trí j nhỏ nhất (j >= i) sao cho đoạn từ i -> j chứa K xâu đối xứng với K độ dài đã cho. Việc tính toán có thể sử dụng quy hoạch động trạng thái: G[t] là vị trí gần i nhất sao cho tồn tại các xâu con đối xứng trong trạng thái t, sử dụng mảng Next để cập nhật cho trạng thái tiếp theo. Nếu xâu thứ x (độ dài l[x]) chưa nằm trong trạng thái t, ta có thể cập nhật G[t’] (t’ = t + 2 ^ (x-1)), sử dụng mảng Next để tìm vị trí gần nhất chứa xâu con đối xứng độ dài l[x].

ADDITION

N <= 10^9 => x, y, z <= 10^6.

Với 40% đầu tiên, N <= 10^6 => x, y, z <= 1000 có thể kiểm tra tất cả các cặp x, y và kiểm tra xem có tồn tại z thoả mãn hay không.

Với 60% còn lại, giả sử a >= b >= c. Nếu c >= 10^6 => x, y, z <= 1000 => sử dụng cách ở trên.
Nếu c <= 10^6, ta đếm số lượng bộ (x, y, z) bằng cách đếm từng cặp (y, z) thoả mãn cho mỗi x.

Với mỗi giá trị của x, ta cần đếm số nghiệm by + cz = N - a * x = N’.

Nhận thấy, nếu N’ chia c dư t => by chia c cũng dư t. Từ đó, ta khởi tạo mảng a[t] chứa tất cả các giá trị b * y (với y trong khoảng 0 đến 10^6) sao cho (b * y) mod c = t.

Với mỗi N’, giả sử t = N’ mod c => b*y mod c = t => đếm số lượng số <= N’ trong mảng a[t]. Với mỗi số như thế ta tìm được duy nhất 1 số z => với mỗi x, N’ = N - ax, t = N’ mod c, đếm số lượng số <= N’ trong mảng a[t].

Với những ai sử dụng C++ có thể dùng a[t] như một vector, nếu dùng pascal các bạn có thể lưu lại các bản ghi a[i].t và a[i].by rồi có thể sử dụng chặt nhị phân để đếm số lượng i có a[i].t = t và a[i].by <= N’.

Mọi thắc mắc về lời giải các bạn/thầy cô có thể gửi vào mail ở trên.