Trại Đông Đà Lạt

Hướng dẫn giải bài tập

29-31/10/2023

Ngày 29 tháng 10 năm 2023

Bài 1. Trò chơi X – SQUARE

Bài 2. Trả nợ ngân hàng – DEBT

Bài 3. Chở vàng – GOLD

Bài 1. Trò chơi X – SQUARE

Bài 2. Trả nợ ngân hàng – DEBT

Bài 3. Chở vàng – GOLD

Square

Trò chơi trên một lưới ô vuông $N \times N$, với N là số lẻ. Có chính xác $2 \times N - 1$ trong số các ô được điền X và phần còn lại để trống (biểu thị bằng ký tự '.'). Trong mỗi nước đi, người chơi có thể chọn và đổi hai hàng hoặc chọn và đổi hai cột. Mục tiêu của trò chơi là làm cho tất cả các ô X nằm trên hai đường chéo chính của lưới, tạo thành hình chữ X lớn và giành chiến thắng, như trong ví dụ sau với $N = 5$:

```
X...X
.X.X.
..X..
.X.X.
X...X
```

Yêu cầu: Cho một trạng thái của lưới ô vuông, hãy xác định xem liệu có thể đưa về trạng thái chiến thắng của trò chơi không?

Subtask 1 ($3 \leq N \leq 5$)

- Một thuật toán trực tiếp là loang BFS tất cả các trạng thái lưới có thể đến khi gặp trạng thái chiến thắng hoặc kết thúc BFS. Có thể làm tốt hơn?
- Nhận xét: Tập hợp các ô trong một hàng cụ thể trong lưới ban đầu sẽ luôn ở cùng nhau, theo một thứ tự nào đó, trong một hàng nào đó. Điều này là do việc hoán đổi cột chỉ thay đổi thứ tự tương đối của các ô đó và hoán đổi hàng chỉ di chuyển toàn bộ tập hợp các ô đó cùng nhau. Vì vậy, các hoạt động trên bảng không thể trao đổi các ô riêng lẻ giữa các hàng khác nhau; chúng chỉ có thể thay đổi thứ tự nội bộ và thứ tự tương đối của các hàng hiện có.
- Do hoán đổi cột là cách duy nhất để thay đổi thứ tự nội bộ trong một hàng, việc thay đổi một hàng để có một thứ tự nhất định sẽ mang lại cho tất cả các hàng có cùng thứ tự nội bộ. Tuy nhiên, ta có thể chọn bất kỳ thứ tự nội bộ nào mong muốn bằng cách chọn cột có ô mà ta muốn đặt đầu tiên và hoán đổi cột đó sang vị trí cột đầu tiên, v.v.

Subtask 1 ($3 \leq N \leq 5$)

- Các nhận xét trên cũng đúng cho cột một cách độc lập: nếu sắp xếp lại các hàng và sau đó sắp xếp lại các cột, thì các thao tác sau sẽ không thay đổi thứ tự đã chọn của các hàng. Vì vậy, ta có thể chọn bất kỳ $N!$ thứ tự có thể có cho N hàng, và bất kỳ $N!$ thứ tự có thể có cho N cột, với tổng số $(N!)^2$ trạng thái khác nhau.
- Có thể áp dụng DFS để thực hiện duyệt theo các nhận xét trên. Độ phức tạp $O((N!)^2)$.

Subtask 2 ($3 \le N \le 55$)

- Nhận xét: để đạt trạng thái chiến thắng, lưới phải có một ô X là X duy nhất trong hàng và trong cột của nó, tạo thành hình chữ thập đơn. Phần còn lại của lưới chứa một tập hợp $N/2$ hình chữ nhật lồng nhau. Mỗi hình chữ nhật được xác định bởi hai hàng và hai cột và có một ô X ở mỗi giao điểm trong số bốn giao điểm của các hàng/cột đó và không có ô X nào ở bất kỳ nơi nào khác.
- Bắt đầu tại trạnh thái chiến thắng và thực hiện một số thao tác hoán đổi. Theo nhận xét trong subtask 1, nếu hai trong số các góc của hình chữ nhật nằm cùng một hàng hoặc trong cùng một cột thì chúng sẽ luôn như vậy qua các phép biến đổi. Vì vậy, các thao tác hoán đổi không thể tạo hoặc hủy các hình chữ nhật, mặc dù chúng có thể thay đổi vị trí tương đối của bốn góc.
- Vì vậy, ta chỉ cần hỏi: Lưới ban đầu có chính xác $N/2$ hình chữ nhật và có chính xác một hình chữ thập đơn không? Nếu không, câu trả lời là IMPOSSIBLE. Trái lại, ta có thể hoán đổi lưới ban đầu về trạng thái chiến thắng như sau. Chọn một hình chữ nhật và thực hiện hoán đổi hàng và cột sao cho các góc của nó trở thành các góc ngoài cùng của hình chữ X lớn hơn. Sau đó giải đệ quy phần còn lại của lưới dưới dạng một bài toán con, v.v.

Subtask 2 ($3 \leq N \leq 55$)

Đơn giản hơn: ta không cần phải thực sự thực hiện bất kỳ hoán đổi nào! Chỉ cần kiểm tra lưới ban đầu để tìm các hình chữ nhật và một chữ thập đơn. Chỉ cần duyệt một lần qua các ô lưới, lần lượt xem xét từng hàng.

- Nếu một hàng không có hoặc nhiều hơn hai ô X, thì nó không thể là một phần của hình chữ nhật hoặc hình chữ thập đơn, kết quả trả về IMPOSSIBLE .
- Nếu một hàng có một ô X, ta xem nó có là một phần của chữ thập đơn không; lưu vị trí cột của X đó và nếu sau đó ta thấy một hàng chéo đơn lẻ khác thì kết quả là IMPOSSIBLE.
- Nếu một hàng có hai ô X (một cặp X), lưu lại các cột của hai ô đó. Khi đã kiểm tra tất cả các hàng, cần kiểm tra để thấy rằng mỗi cặp X này ghép theo cột với chính xác một cặp X khác và không ghép với bất kỳ cặp nào khác. Hơn nữa, vị trí của chữ X trong chữ thập đơn là cột duy nhất không được sử dụng bởi bất kỳ cặp nào khác. Nếu đúng thì kết quả trả về POSSIBLE.

Độ phức tạp: $O(N^4)$.

Bài 1. Trò chơi X – SQUARE

Bài 2. Trả nợ ngân hàng – DEBT

Bài 3. Chở vàng – GOLD

Trả nợ ngân hàng – DEBT

N ngân hàng đến gặp Thanh cùng với N bản hợp đồng, bản hợp đồng thứ i cho biết anh ấy đang nợ ngân hàng thứ i số tiền là a_i đồng. Luật sư đại diện của N ngân hàng đã xếp N bản hợp đồng thành một hàng ngang trên bàn từ trái sang phải là hợp đồng thứ nhất cho đến hợp đồng thứ N, ít nhất trong hôm nay Thanh phải trả tiền cho các bản hợp đồng xếp ở vị trí thứ $L, L+1, \ldots, R$ trên bàn. May mắn thay, luật sư là một người bạn của Thanh nên anh ấy cho phép Thanh được đổi vị trí của hợp đồng thứ i và hợp đồng thứ j trên bàn với nhau, một lần thực hiện đổi như vậy cần $|i-j|$ phút. Vì tính chất công việc nên Thanh chỉ được cho K phút để thực hiện. Hãy giúp Thanh tính tổng số tiền tối thiểu phải trả trong hôm nay nếu như Thanh thực hiện chuyển đổi vị trí giữa các hợp đồng trong thời gian không quá K phút.

Subtask 2 ($N \leq 50; R = N$): $O(N^2 * K)$

Chia dãy ban đầu thành 2 dãy $a_1, a_2, \ldots, a_{L-1}$ và dãy $a_L, a_{L+1}, \ldots, a_N$. Ta có:

- Một lần thực hiện đổi thì ta đổi 1 hợp đồng trong dãy thứ nhất với 1 hợp đồng trong dãy thứ hai.
- Đối với một hợp đồng bất kỳ, ta chỉ quan tâm nó có được chọn ra để thực hiện đổi hay không, chứ không cần quan tâm tới việc nó được chọn để đổi với hợp đồng nào. Bởi vì khi ta chọn 1 tập các hợp đồng có chỉ số $\{i_1, i_2, \ldots, i_t\}$ trong dãy thứ nhất để đổi với 1 tập các hợp đồng có chỉ số $\{j_1, j_2, \ldots, j_t\}$ trong dãy thứ hai thì chi phí luôn là: $(j_1 + j_2 + ... + j_t) - (i_1 + i_2 + ... + i_t)$.

Subtask 2 ($N \leq 50; R = N$): $O(N^2 * K)$

Gọi $dp[i, j, k]$ là số tiền tối thiểu phải trả khi xét đến hợp đồng thứ i ($i = 1 \leftarrow L - 1$) trong dãy thứ nhất và hợp đồng thứ j ($j = L \leftarrow N$) trong dãy thứ hai với thời gian tối đa là k. Do $R = N$ nên ta chỉ cần thực hiện qhđ từ một phía vì ta biết là chỉ có đổi hợp đồng i ($1 \leq i < L$) và hợp đồng j ($L \leq j \leq N$).

- $dp[i, j, k] = \min(dp[i-1, j, k], dp[i, j-1, k], dp[i, j, k-1], dp[i-1, j-1, k-(j-i)] + a_i - a_j)$.

Kết quả bài toán là $\max(dp[L-1, N, K])$.

Subtask 3 ($N \leq 50$): $O(N^2 * K)$

Chia dãy ban đầu thành 3 dãy $a_1, a_2, \ldots, a_{L-1}$, dãy $a_L, a_{L+1}, \ldots, a_R$ và dãy $a_{R+1}, a_{R+2}, \ldots, a_N$. Ta phải thực hiện qhđ từ 2 phía, xác định một điểm P ở giữa tối ưu ($L \leq P < R$) sao cho:

- Hợp đồng i ($1 \leq i < L$) sẽ chỉ đổi với các hợp đồng i' ($L \leq i' \leq P$);
- Hợp đồng j ($R < j \leq N$) sẽ đổi vs các hợp đồng j' ($P < j' \leq R$).

Gọi:

- $dpL[i, j, k]$ là số tiền tối thiểu phải trả khi xét đến hợp đồng thứ i ($i = 1 \leftarrow L - 1$) trong dãy thứ nhất và hợp đồng thứ j ($j = L \leftarrow R$) trong dãy thứ hai với thời gian tối đa là k.
- $dpR[i, j, k]$ là số tiền tối thiểu phải trả khi xét đến hợp đồng thứ i ($i = N \leftarrow R + 1$) trong dãy thứ ba và hợp đồng thứ j ($j = R \leftarrow L$) trong dãy thứ hai với thời gian tối đa là k.

Subtask 3 ($N \leq 50$): $O(N^2 * K)$

Nhận xét: Trong dãy thứ hai, với hai hợp đồng x, y ($x < y$) được chọn để đổi thì trường hợp dùng hợp đồng x đổi với 1 hợp đồng trong dãy thứ nhất và hợp đồng y đổi với 1 hợp đồng trong dãy thứ ba chắc chắn tối ưu hơn trường hợp dùng hợp đồng x đổi với 1 hợp đồng trong dãy thứ ba và hợp đồng y đổi với 1 hợp đồng trong dãy thứ nhất.
Từ đó suy ra tồn tại một vị trí i trong dãy thứ hai sao cho các hợp đồng trước hợp đồng thứ i sẽ chỉ đổi với các hợp đồng trong dãy thứ nhất, còn các hợp đồng từ hợp đồng i về sau chỉ đổi với các hợp đồng trong dãy thứ ba. Vì vậy ta cập nhập kết quả:

- $Ans = \max(Ans, dpL[L-1, i-1, T] + dpR[R+1, i, K-T]$.

Subtask 4 ($N \leq 100$): $O(N^2 * K)$

Ở subtask này ta cần phải tối ưu bộ nhớ và nhận thấy rằng giá trị $K \leq N^2/4$. Nhờ đó ta có thể tối ưu độ phức tạp thời gian và độ phức tạp không gian.

Bài 1. Trò chơi X – SQUARE

Bài 2. Trả nợ ngân hàng – DEBT

Bài 3. Chở vàng – GOLD

Chở vàng – GOLD

N thị trấn được nối với nhau qua $N-1$ con đường nối liên thông mạng. Con đường thứ i có độ dài là D_i nối 2 thị trấn A_i và B_i. Mức phí của con đường thứ i là T_i.
Nếu hiện tại xe đang chứa X thỏi vàng và đi được 1 đơn vị quãng đường thì sử dụng hết X đơn vị nguyên liệu. Chuyến đi thứ j yêu cầu từ thị trấn A_j mang đến thị trấn B_j được G thỏi vàng (G không đổi).
Bạn được cung cấp chính xác thứ tự tất cả các chuyến đi và các sự kiện thay đổi mức phí của các con đường, hãy tính mức nhiên liệu tối ưu phải sử dụng với mỗi chuyến đi, kết quả chia lấy dư cho 10^9+7.

Subtask 1: Không có sự kiện thay đổi mức phí, $T_1 = T_2 = ... = T_{N-1} = 0$, mỗi thị trấn có tối đa 2 con đường nối đến.

Đồ thị của ta là một đường thẳng, câu trả lời cho từng chuyến đi chính là $G * dist(A, B)$ với $dist(A, B)$ là khoảng cách giữa thị trấn A và thị trấn B. Ta sử dụng mảng cộng dồn để tính giá trị $dist(A, B)$.
ĐPT: $O(N + Q)$.

Subtask 2: Không có sự kiện thay đổi mức phí, $T_1 = T_2 = ... = T_{N-1} = 0$

Câu trả lời cho từng chuyến đi vẫn là $G * dist(A, B)$. Để tính $dist(A, B)$ ta sử dụng LCA.
ĐPT: $O((N + Q) * \log N)$.

Subtask 3: Không có sự kiện thay đổi mức phí, $T_1 = T_2 = ... = T_N, D_1 = D_2 = ... = D_{N-1} = 1$.

Đặt $X = T_1 = T_2 = ... = T_N$ và $M = dist(A, B)$.
Vì $D_1 = D_2 = ... = D_{N-1} = 1$ nên từ thị trấn A đến thị trấn B ta phải đi qua M con đường và mỗi lần đi qua một con đường ta đều trả phí là X thỏi vàng. Như vậy số thỏi vàng ban đầu phải chở là $G + X * M$. Từ đó ta tính được:

- Khi đi qua con đường thứ nhất cần $G + X * (M - 1)$ nhiên liệu.
- Khi đi qua con đường thứ hai cần $G + X * (M - 2)$ nhiên liệu.
- ...
- Khi đi qua con đường thứ M cần G nhiên liệu.

Tổng số nhiên liệu sử dụng là $G * M + X * (M * (M - 1))/2$.
ĐPT: $O((N + Q) * \log N)$.

Subtask 4: Không có sự kiện thay đổi mức phí, mỗi thị trấn có tối đa 2 con đường nối đến

Đồ thị của ta là một đường thẳng, ta chọn đỉnh nằm một đầu của đường thẳng làm gốc gọi là *Root*. Ta xét trường hợp A gần *Root* hơn B. Xem quãng đường đi từ A đến B đi qua m con đường là một dãy các thị trấn $x_0, x_1, \ldots, x_m$ trong đó $x_0 = A, x_m = B$. Đặt t_i, d_i $(1 \leq i \leq m)$ là chi phí và chiều dài của con đường nối thị trấn x_{i-1} với thị trấn x_i trong quãng đường từ A đến B. Tại con đường thứ m:

- Nhiên liệu để đi qua con đường này là $G * d_m$.
- Số thỏi vàng khi chưa trả phí cho con đường này là $G + t_m$.

Tại con đường thứ $m - 1$:

- Nhiên liệu để đi qua con đường này là $(G + t_m) * d_{m-1}$.
- Số thỏi vàng khi chưa trả phí cho con đường này là $G + t_m + t_{m-1}$.

... Tại con đường thứ 1:

- Nhiên liệu để đi qua con đường này là $(G + t_m + ... + t_2) * d_1$.
- Số thỏi vàng khi chưa trả phí cho con đường này là $G + t_m + t_{m-1} + ... + t_1$.

Subtask 4: Không có sự kiện thay đổi mức phí, mỗi thị trấn có tối đa 2 con đường nối đến

Tổng nhiên liệu phải sử dụng để đi từ A đến B là:

$$G * d_m + (G + t_m) * d_{m-1} + ... + (G + t_m + ... + t_2) * d_1$$

$$= G*(d_m+...+d_1)+t_m*(d_{m-1}+...+d_1)+t_{m-1}*(d_{m-2}+...+d_1)+...+t_2*d_1$$

$$= G*dist(A,B)+t_m*dist(A,x_{m-1})+t_{m-1}*dist(A,x_{m-2})+...+t_2*dist(A,x_1)$$

$$= G * dist(A,B) + \sum_{i=2}^{m} t_i * dist(A, x_{i-1})$$

$$= G*[dist(Root,B)-dist(Root,A)]+\sum_{i=2}^{m} t_i*[dist(Root,x_{i-1})-dist(Root,A)]$$

$$= G*[dist(Root,B)-dist(Root,A)]+\sum_{i=2}^{m} t_i*dist(Root,x_{i-1})-\sum_{i=2}^{m} t_i*dist(Root,$$

Các giá trị $\sum_{i=2}^{m} t_i * dist(Root, x_{i-1}), \sum_{i=2}^{m} t_i * dist(Root, A)$ ta sử dụng mảng cộng dồn để tính toán.

Trường hợp B gần *Root* hơn A làm tương tự.

ĐPT: $O(N + Q)$.

Subtask 5: Mỗi thị trấn có tối đa 2 con đường nối đến.

Tương tự subtask 4, trước hết ta tính trước khoảng cách từ $Root$ đến các đỉnh khác. Khi thay đổi chi phí con đường thứ i từ t_i thành t_i', ta sử dụng 2 cây segment tree, trong đó:

- Cây thứ nhất, tại nút lá quản lý con đường thứ i, ta update giá trị của nó từ $t_i * dist(Root, x_{i-1})$ thành $t_i' * dist(Root, x_{i-1})$.
- Cây thứ hai, tại nút lá quản lý con đường thứ i, ta update giá trị của nó từ t_i thành t_i'.

Để tính các giá trị
$\sum_{i=2}^{m} t_i * dist(Root, x_{i-1}), \sum_{i=2}^{m} t_i * dist(Root, A)$ thì ta dựa vào 2 cây segment tree này.
ĐPT: $O((N+Q) * \log N)$.

Subtask 6: $N, Q \leq 5000$

Tại mỗi chuyến đi ta có thể thực hiện DFS từ thị trấn B về thị trấn A để tính được số vàng ban đầu phải chở, từ đó DFS từ thị trấn A đến thị trấn B để tính số nhiên liệu sử dụng.
ĐPT: $O(N * Q)$.

Subtask 7: $N, Q \leq 10^5$

Gọi C là cha chung gần nhất của A và B. Từ đó quãng được từ A đến B sẽ chia thành 2 giai đoạn đó là từ A đến C và từ C đến B.
Xét quãng đường đi từ C đến B để có G thỏi vàng tại B thì ta có thể tính được lượng nhiên liệu sử dụng theo subtask 5 và đồng thời tính được số thỏi vàng ở C, gọi là G'.
Xét quãng đường đi từ A đến C để có G' thỏi vàng tại C thì ta có thể tính được lượng nhiên liệu sử dụng theo subtask 5.
ĐPT: $O((N + Q) * \log N)$.