Hướng Dẫn Giải
Đề Thi Thử VOI2023 Ngày 2

TỔNG QUAN ĐỀ THI

Tên bài	**Tên chương trình**	**Dữ liệu đầu vào**	**Kết quả đầu ra**	**Số điểm**
Bài 4. Bộ bài cân bằng	KTURN.*	KTURN.INP	KTURN.OUT	7 điểm
Bài 5. Hành trình leo núi	DINCPATH.*	DINCPATH.INP	DINCPATH.OUT	7 điểm
Bài 6. Thám hiểm vũ trụ	ENGINE.*	ENGINE.INP	ENGINE.OUT.	6 điểm

Dấu * được thay thế bởi `pas` hoặc `cpp` của ngôn ngữ lập trình tương ứng là Pascal và C++.

Mục lục

Bài 4. Bộ bài cân bằng — KTURN

Hướng dẫn giải

Subtask 1: 15% số test ứng với $k = 0$.

Là bài toán cổ điển: "Tìm đoạn con liên tiếp có tổng lớn nhất".

ĐPT: $O(N)$.

Subtask 2: 15% số test ứng với $k = 1$ và $n \leq 5000$

Giả sử đoạn con có tổng lớn nhất trên dãy A sau khi chỉnh sửa là $[\ell, r]$ thì thực tế trước đó ta chỉ cần chỉnh sửa trên đoạn $[u, v]$ thỏa mãn $\ell \leq u \leq v \leq r$.

Gọi:

- $End[i]$ = giá trị (tổng) đoạn con có tổng lớn nhất kết thúc tại A_i.
- $Start[i]$ = giá trị (tổng) đoạn con có tổng lớn nhất bắt đầu tại A_i.

Sau đó duyệt cặp (u, v) mà $u \leq v$ và so sánh kết quả bài toán với $End[u-1] + Start[v+1] + SUM(u, v)$ trong đó $SUM(u, v)$ có thể tính nhanh được là giá trị $A_u * B_u + A_{u+1} * B_{u+1} + ... + A_v * B_v$ trong $O(1)$.

ĐPT: $O(N^2)$.

Subtask 3: 20% số test ứng với $k = 1$.

Dựa trên ý tưởng của subtask 2, xem phần bên trái vị trí u là phần 1, đoạn $[u, v]$ là phần thứ 2, đoạn bên phải v là phần thứ 3.

Từ đó áp dụng quy hoạch động. Gọi $dp[i, t]$ là giá trị lớn nhất của đoạn con kết thúc tại A_i mà tại đó A_i đóng vai trò là phần thứ t. Mặc định hiểu nếu $t = 0$ thì xem như phần tử đó không được chọn và do đó $dp[i, 0] = 0$.

Ta có công thức như sau:

- $dp[i, 1] = A_i + \max(0, dp[i-1, 1])$.
- $dp[i, 2] = A_i * B_i + \max(0, dp[i-1, 1], dp[i-1, 2])$.
- $dp[i, 3] = A_i + \max(0, dp[i-1, 2], dp[i-1, 3])$.

Dễ thấy kết quả là $max(dp[i, 1], dp[i, 2], dp[i, 3])$ với mọi $i \in [1, N]$.

ĐPT: $O(N)$.

Subtask 4: 25% số test ứng với $k = 2$.

Tương tự như Subtask 3, chỉ có điều ta phải chia thành hai trường hợp là hai đoạn được chọn giao nhau hay không, từ đó ta có thể biết được đoạn được chọn được chia thành bao nhiêu phần, mỗi phần cần phải thay đổi giá trị như thế nào.

ĐPT: $O(N)$.

Subtask 5: 25% số test còn lại ứng với $k \leq 10$.

Chúng ta sử dụng phương pháp quy hoạch động "đóng-mở ngoặc". Gọi $dp[i, open, close]$ là khi ta xét đến vị trí thứ i, trước đó có $open$ đoạn đang mở và $close$ đoạn đã đóng. Khi đó giả sử muốn update lên và chọn A_i thì giá trị A_i lúc đó sẽ là $A_i * B_i^{open}$. Chúng cần kiểm soát để $open + close \leq K$. Kết quả cuối cùng sẽ là $\max(dp[n, 0, close])$ với $close \leq K$.

Chi tiết cách chuyển trạng thái được thực hiện trong code mẫu.

ĐPT: $O(N * K^3)$.

Bài 5. Hành trình leo núi — DINCPATH

Subtask 1: 20% số test ứng với $N \leq 20$.

Nhận xét: Từ điều kiện $0 < A_{P_2} - A_{P_1} < A_{P_3} - A_{P_2} < ... < A_{P_k} - A_{P_{k-1}}$ ta suy ra được $0 < A_{P_1} < A_{P_2} < \ldots < A_{P_K}$, hay các đỉnh trên đường đi có giá trị tăng dần.

Ở Subtask này, do số đỉnh rất bé ($N \leq 20$) mà số cạnh rất lớn ($M \leq 3 \times 10^5$) nên có thể có rất nhiều cạnh đi qua hai đỉnh. Gọi $numEdge[u][v]$ là số cạnh đi qua hai đỉnh u và v.

Vì $N \leq 20$ nên ta có thể duyệt qua tập các đỉnh và kiểm tra xem đường đi có phải là đường đi dài nhất hay không. Nếu đường đi là đường đi hợp lệ và dài nhất thì ta cộng vào kết qủa một lượng là tích số cạnh giữa các cặp đỉnh liên tiếp.

Độ phức tạp: $\mathcal{O}(2^N \times N)$.

Subtask 2: 20% số test ứng với $N \leq 500$.

Gọi $dp[u][v]$ là số đường đi dài nhất hợp lệ dài nhất bắt đầu từ cạnh nối giữa hai đỉnh u và v theo chiều từ u đến v. Ta có 2 trường hợp:

- Nếu $A_u < A_v$: $dp[u][v] = numEdge[u][v] \times \Sigma_{k=1}^{N} dp[v][k](A_v - A_u < A_k - A_v)$
- Nếu $A_u \geq A_v$: $dp[u][v] = 0$

Ta tính số đường đi song song với việc tính hàm dp.

Độ phức tạp: $\mathcal{O}(N^3)$.

Subtask 3: 20% số test ứng với các ngọn núi xếp thành một dãy núi thẳng hàng với nhau, nói cách khác đồ thị tương ứng là một mạch thẳng.

Ta trải đồ thị thành mảng và tìm đường đường đi dài nhất liên tiếp thoả mãn và số lượng đường đi dài nhất thoả mãn.

Độ phức tạp: $\mathcal{O}(N)$.

Subtask 4: 20% số test khác ứng với $M = N - 1$, và các đỉnh trong đồ thị liên thông với nhau.

Ý tưởng như Subtask 5 nhưng dễ nhận ra và dễ cài đặt hơn.

Subtask 5: 20% còn lại ứng với $N \leq 3 \times 10^5, M \leq 5 \times 10^5$.

Nhận xét: Với cạnh (u, v), nếu $A_u \le A_v$ thì ta chỉ có thể đi từ u đến v chứ không thể đi từ v đến u. Ta định chiều cạnh để một cạnh có hướng đi từ đỉnh có trọng số bé hơn đi đến đỉnh có trọng số lớn hơn, trọng số của cạnh là hiệu trọng số đỉnh lớn hơn trừ trọng số đỉnh bé hơn, gọi là $w(u, v)$.

Do một cạnh có hướng đi từ đỉnh có trọng số bé hơn đi đến đỉnh có trọng số lớn hơn nên đồ thị là một DAG (Directed Acyclic Graph - Đồ thị có hướng không có chu trình). Nên ta có thể sử dụng kỹ thuật quy hoạch động trên DAG cho bài này.

Ta sắp xếp các cạnh đi ra của 1 đỉnh theo thứ tự giảm dần. Duyệt các cạnh (u, v) theo thứ tự trọng số đỉnh v giảm dần, ta có $dp[(u, v)] = 1 + \max dp[(v, k)]$ với $w(u, v) < w(v, k)$. Ta tính số đường đi song song với việc tính hàm dp.

Sử dụng cấu trúc dữ liệu **Fenwick tree** để tối ưu việc tính toán.

Độ phức tạp: $\mathcal{O}(N \times \log_2(N))$.

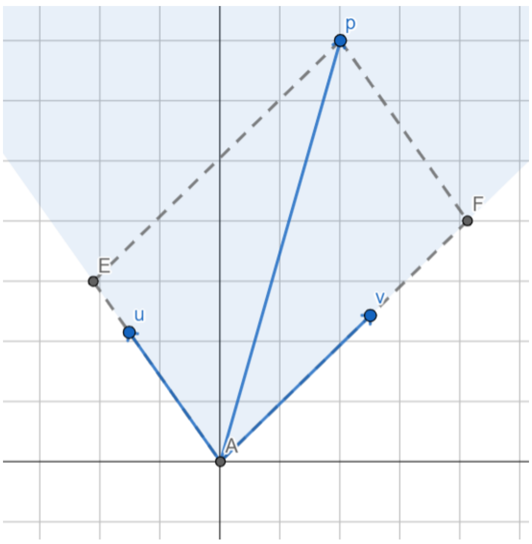
Bài 6. Thám hiểm vũ trụ — ENGINE

Hướng dẫn giải

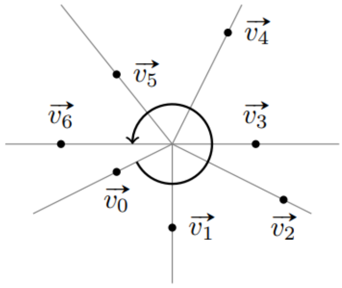
Subtask 1: 30% số test ứng với $n \le 16$.

Nhận xét 1: Với hai vector khác phương bất kì:

- Mọi vector nằm GIỮA phần góc nhỏ hơn 180 độ đều CÓ thể được biểu diễn bởi tổ hợp tuyến tính không âm của hai vector.
- Mọi vector nằm NGOÀI phần góc nhỏ hơn 180 độ đều KHÔNG thể được biểu diễn bởi tổ hợp tuyến tính không âm của hai vector.



Giả sử các vector đã được polar sort theo ngược chiều kim đồng hồ (tức là sắp xếp theo thứ tự va chạm với một tia xuất phát từ góc tọa độ và quay ngược chiều kim đồng hồ).



Nhận xét 2: Một tập vector là hợp lệ khi và chỉ khi không có hai vector liên tiếp nào mà góc ngược chiều kim đồng hồ giữa chúng lớn hơn hoặc bằng 180 độ.

Từ nhận xét trên, ta sinh dãy nhị phân để tìm tập vector hợp lệ có tổng chi phí nhỏ nhất.

Subtask 2: 20% số test ứng với $n \leq 50$

Nhận xét 3: Luôn tồn tại tập vector hợp lệ có số lượng phần tử nhỏ hơn hoặc bằng 4.

Chứng minh: Với một tập vector t có 5 phần tử trở lên, sẽ luôn tồn tại hai vector t_k, t_{i+2} mà góc ngược chiều kim đồng hồ giữa chúng nhỏ hơn 180 độ. Do đó ta có thể bỏ đi vector t_{i+1} và tập vector vẫn hợp lệ.

Với nhận xét trên, ta chỉ cần duyệt qua các tập gồm 3 hoặc 4 vector.

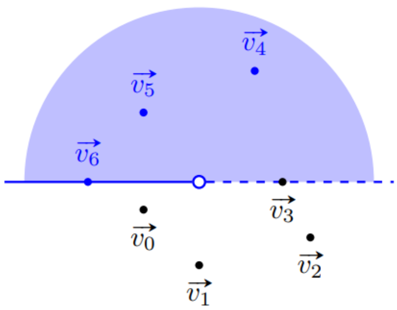
Subtask 3: 30% số test ứng với $n \leq 2000$

Nhận xét 4: Với các tập vector hợp lệ có 4 phần tử, ta chỉ cần xét các tập mà hai vector liên tiếp luôn tạo thành góc 90 độ (trong trường hợp ngược lại, luôn có cách bỏ đi một vector sao cho tập vẫn hợp lệ).

Việc xét các tập trên có thể được thực hiện trong $O(n \log n)$ bằng việc tối giản tọa độ của các vector và cấu trúc dữ liệu map.

Để xét các tập hợp lệ gồm 3 vector một cách hiệu quả, ta chia các vector thành hai tập:

- Tập X gồm các vector có $b_i > 0$ hoặc ($b_i = 0$ và $a_i < 0$)
- Tập Y gồm các vector còn lại



Nhận xét 5: Với các tập hợp lệ gồm 3 vector, sẽ có một vector thuộc tập X, hai vector thuộc tập Y hoặc ngược lại.

Xét trường hợp tập cần tìm chứa một vector thuộc tập X, hai vector thuộc tập Y (trường hợp còn lại tương tự). Gọi vector thuộc tập X được xét đến là p. Khi đó, trong hai vector thuộc tập Y sẽ có một vector nằm bên trái đường thẳng chứa vector p và một vector nằm bên phải đường thẳng đó.

Do đó, với mỗi vector p thuộc tập X, ta cần tìm vector có chi phí nhỏ nhất nằm bên trái đường thẳng chứa vector p, và vector có chi phí nhỏ nhất nằm bên trái đường thẳng chứa vector p. Việc tìm hai vector trên có thể được hiện bằng cách duyệt trâu trong $O(n)$.

Độ phức tạp: $O(n^2)$

Subtask 4: 20% số test còn lại ứng với $n \leq 2 \times 10^5$.

Dùng kĩ thuật sweep line kết hợp two pointer để tính nhanh vector thuộc tập Y có chi phí nhỏ nhất nằm lần lượt bên trái và bên phải đường thẳng chứa vector p.

Độ phức tạp: $O(n \log n)$ do chi phí sắp xếp.