KUPC2020 autumn B

writer: etonagisa

2020 年 10 月 10 日

紙の番号の小さいほうから順に整数を選んでいくことを考えます。i 番目までの紙の整数の選び方を確定したとすると、$i+1$ 番目以降の紙での整数の選び方は、i 番目の紙で選んだ整数にのみ依存することがわかります。そこで次のような動的計画法を考えます。

$dp[i][j] = i$ 番目の紙までの整数の選び方を確定し、i 番目の紙では $v_{i,j}$ を選んで単調増加な数列を作る方法の数

定義から $dp[1][j] = 1 (1 \leq j \leq K)$ です。また $i > 1$ に対して、$dp[i][j] = \sum_{v_{i-1,l} \leq v_{i,j}} dp[i-1][l]$ が成り立ちます。この式を素直に実装すると計算量は $O(NK^2)$ となり、実行時間制限に間に合いません。

$dp[i-1]$ の値が全てわかっている状態から、$dp[i]$ の値を $O(K)$ で全て計算することを考えます。これができれば全体の計算量は $O(NK)$ となり間に合います。各紙に書かれている整数がソートされていることから、各 $j(1 \leq j \leq K)$ に対してある整数 l_j が存在し、$dp[i][j] = \sum_{l \leq l_j} dp[i-1][l]$ が成り立ちます。直観的には、l_j は $v_{i-1,l} \leq v_{i,j}$ が成り立つ最大の l です。さらにこの l_j は j について広義単調増加します。したがって j と l_j についてしゃくとり法を行うことで、$dp[i]$ の値が $O(K)$ で計算できました。

別解として、あらかじめ $dp[i-1]$ の累積和をとっておき、l_j を二分探索を使って毎回求めることでも正しい解が得られます。この場合の計算量は $O(NK \log K)$ ですが、十分間に合います。