KUPC2020 spring D：Xor Array 解説

writer: heno239

2020 年 3 月 20 日

解法 1

以下、 a と b の排他的論理和を $a \oplus b$ と表すことにします。
$dp[i][j][k]$:= 長さ i の数列で、最後の要素が j 、全ての要素の xor が k となる場合の数、と定義します。

数列の最後に $j+1$ を何個か置くことを考えると、dp の遷移として

$$\begin{cases} dp[i+l][j+1][k] = dp[i+l][j+1][k] + dp[i][j][k] & (l \text{ は偶数}) \\ dp[i+l][j+1][k \oplus (j+1)] = dp[i+l][j+1][k \oplus (j+1)] + dp[i][j][k] & (l \text{ は奇数}) \end{cases}$$

が成立します。これを配る側から貰う側に書き直すと、

$$dp[i][j][k] = \left(\sum_{0 \le l \le i, l \text{ は偶数}} dp[i-l][j-1][k] \right) + \left(\sum_{0 \le l \le i, l \text{ は奇数}} dp[i-l][j-1][k \oplus x] \right)$$

が成立します。

累積和を用いて高速化することで、$O(NX^2)$ でこの問題が解けます。

解法 2

dp の定義は解法 1 と同じですが，今度はまず j を固定して先に i, k を動かして dp テーブルを更新します。

$$\begin{cases} dp[i+1][j][k \oplus j] = dp[i+1][j][k \oplus j] + dp[i][j][k] & (j \text{ を置く場合}) \\ dp[i][j+1][k] = dp[i][j+1][k] + dp[i][j][k] & (\text{もう } j \text{ は置かない場合}) \end{cases}$$

が成立し、$O(NX^2)$ でこの問題が解けます。