J- Range Xor Query

原案・解説 TAB

2023 年 5 月 4 日

この問題は平方分割により解くことができます。長さ N の数列 A を B 個ごとのブロックに分割し、ブロックごとに binary trie[*1]を持っておきます。各クエリは以下のように処理することができます。

■クエリ 1　k 番目の値が所属するブロックの binary trie から A_k を削除し、$A_k \oplus x$ を追加する。A_k を $A_k \oplus x$ で更新する。($\mathrm{O}(\log \max(A))$)

■クエリ 2　区間 $[l, r]$ に完全に含まれるブロックについては、binary trie を用いて $A_i \oplus x$ の最小値を取得する。$[l, r]$ と一部が重なっているブロックについては、数列 A を見て愚直に $\mathrm{O}(B)$ で最小値を求める。($\mathrm{O}(B + \frac{N}{B} \log \max(A))$)

$B = \sqrt{N \log \max(A)}$ とすることで、全体で $\mathrm{O}(Q\sqrt{N \log \max(A)})$ で解くことができます。

別解

binary trie の各頂点に、平衡二分探索木[*2]を持たせてインデックスを管理しておく方針でも解くことができます。

binary trie の各葉には、その葉が対応する値が a だとすると、$A_i = a$ となるような i を追加します。葉以外の頂点は、子が持つ要素を全て追加します。こうすることで、ある頂点を根とした部分木の葉に対応する値が、数列 A の区間 $[l, r]$ に存在するかどうかを二分探索で求めることができます。[*3]binary trie の根から、区間 $[l, r]$ に存在する値で、より小さい方に進むことを繰り返すことでクエリ 2 に答えることができます。

計算量は $\mathrm{O}((N + Q) \log N \log \max(A))$ です。

コメント

領域木[*4]を使った $\mathrm{O}((N + Q) \log N \log \max(A))$ の解法はメモリ制限を超えてしまうみたいです。制限が厳しくて申し訳ありません。

また、テストケースが弱く、別解のようにインデックスを管理し、答えを小さい方から順番に、存在するか確認していく方針が通ってしまっていたようです。

[*1] 追加する要素の最大値を M として、要素の追加・削除、x と xor した値の最小値の取得、などが $\mathrm{O}(\log M)$ でできるデータ構造です。

[*2] C++ なら std::set で十分

[*3] l 以上の最小値 m を見つけ、$m \leq r$ かどうかを見れば良いです。

[*4] セグメントツリーの各頂点に binary trie を持たせる